Tissue P systems can be simulated efficiently with counting oracles[⋆]

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 prove that polynomial-time tissue P systems with cell division or cell separation can be simulated efficiently by Turing machines with oracles for counting problems. This shows that the corresponding complexity classes are included in $\mathbf{P}^{\#\mathbf{P}}$, thus improving, under standard complexity theory assumptions, the previously known upper bound **PSPACE**.

1 Introduction

Tissue P systems [4] are known to solve **NP**-complete (and **coNP**-complete) problems in polynomial time when cell division [9] or cell separation rules [6] are available in addition to the standard, context-sensitive communication rules. In terms of complexity classes, this is denoted by $\mathbf{NP} \cup \mathbf{coNP} \subseteq \mathbf{PMC}_{\mathcal{TDC}}$ and $\mathbf{NP} \cup \mathbf{coNP} \subseteq \mathbf{PMC}_{\mathcal{TSC}}$, respectively. Division and separation rules allow the creation of exponentially many cells in polynomial time; the difference is that division replicates the contents of the original cell, while separation distributes such contents between the resulting cells according to the nature of the objects.

The previously known upper bound to the classes of problems solved in polynomial time by tissue P systems with cell division [11] or separation [10] is **PSPACE**, a class of problems also solved by P systems with active membranes [1]. Unlike these, tissue P systems lack a complex hierarchical membrane structure, a limitation they share with P systems with elementary active membranes, where membranes containing further membranes cannot divide; the problems solved by the latter are known to be bounded by $\mathbf{P}^{\#\mathbf{P}}$ [3], a class conjecturally smaller than **PSPACE**.

In this paper we show that the $\mathbf{P}^{\#\mathbf{P}}$ upper bound also applies to tissue P systems with cell division or cell separation; we describe a simulation that runs in polynomial time by delegating the communication between regions to an oracle for a counting problem.

[⋆] This work was partially supported by Università degli Studi di Milano-Bicocca, FA 2013: “Complessità computazionale in modelli di calcolo bioispirati: Sistemi a membrane e sistemi di reazioni”.

2 Basic notions

We begin by recalling the definition of tissue P systems with division and separation rules; for a more detailed introduction on multiset processing and tissue P systems, we refer the reader to the original paper [4].

Definition 1. *A* tissue P system *is a structure $\Pi = (\Gamma, E, w_1, \ldots, w_d, R)$, where:*

- *Γ is an alphabet, i.e., a finite non-empty set of symbols, usually called* objects*;*
- *$E \subseteq \Gamma$ is the alphabet of objects initially located in the external environment, in* infinitely *many copies;*
- *$d \geq 1$ is the* degree *of the system, i.e., the initial number of cells;*
- *$w_1, \ldots, w_d$ are* finite *multisets over Γ, describing the initial contents of the d cells; here $1, \ldots, d$ are* labels *identifying the cells of the P systems, and 0 is the label of the external environment;*
- *R is a finite set of rules.*

The rules of R are of the following types:

(a) *Communication rules*, denoted in this paper by $[u]_h \leftrightarrow [v]_k$ and in the literature by $(h, u/v, k)$, where h and k are distinct labels (including the environment), and u and v are multisets over Γ (at least one of them nonempty): these rules are applicable if there exists a region with label h containing u as a submultiset and a region k containing v as a submultiset; the effect of the rule is to exchange u and v between the two regions. If $h = 0$ (resp., $k = 0$) then u (resp., v) must contain at least an object from $\Gamma - E$, i.e., an object with finite multiplicity[1]. In this paper we consider a rule $[u]_h \leftrightarrow [v]_k$ and its syntactic reverse $[v]_k \leftrightarrow [u]_h$ to be the same rule.

(b) *Division rules*, of the form $[a]_h \to [b]_h \; [c]_h$, where $h \neq 0$ is a cell label and $a, b, c \in \Gamma$: these rules can be applied to a cell with label h containing at least one copy of a; the effect of the rule is to divide the cell into two cells, both with label h; the object a is replaced in the two cells by b and c, respectively, while the rest of the original multiset contained in h is replicated in both cells.

(c) *Separation rules*, of the form $[a]_h \to [\Gamma_1]_h \; [\Gamma_2]_h$, where $h \neq 0$ is a cell label, $a \in \Gamma$, and $\{\Gamma_1, \Gamma_2\}$ is a partition of Γ: these rules can be applied to a cell with label h containing at least one copy of a; the effect of the rule is to separate the cell into two cells, both with label h; the object a is consumed, while the objects from Γ_1 in the original multiset contained in h are placed inside one of the cells, and those from Γ_2 in the other. All separation rules in R must share the same partition $\{\Gamma_1, \Gamma_2\}$ of Γ.

A *tissue P system with cell division* only uses communication and division rules, while a *tissue P system with cell separation* only uses communication and separation rules.

[1] Since communication rules are applied in a maximally parallel way, this restriction avoids the situation where infinitely many objects from the environment simultaneously enter a cell.

A *configuration* $\mathcal{C}$ of a tissue P system consists of a multiset over $\Gamma - E$ describing the objects appearing with finite multiplicity in the environment, and a multiset of pairs (h, w), where h is a cell label and w a finite multiset over Γ, describing the cells. A *computation step* changes the current configuration according to the following set of principles:

- Each object can be subject to at most one rule, and each cell can be subject to *any number* of communication rules or, *alternatively*, a *single* division or separation rule.
- The application of rules is *maximally parallel*: each region is subject to a maximal multiset of rules (i.e., no further rule can be applied).
- When several conflicting rules can be applied at the same time, a nondeterministic choice is performed; this implies that, in general, multiple possible configurations can be reached after a computation step.

A *halting computation* $\boldsymbol{\mathcal{C}} = (\mathcal{C}_0, \ldots, \mathcal{C}_k)$ of the tissue P system Π is a finite sequence of configurations, where $\mathcal{C}_0$ is the initial configuration, every $\mathcal{C}_{i+1}$ is reachable from $\mathcal{C}_i$ via a single computation step, and no rules are applicable in $\mathcal{C}_k$.

Tissue P systems can be used as language *recognisers* by employing two distinguished objects *yes* and *no*: we assume that all computations are halting, and that either *yes* or object *no* (but not both) is released into the environment, and only in the last computation step, in order to signal acceptance or rejection, respectively. If all computations starting from the same initial configuration are accepting, or all are rejecting, the tissue P system is said to be *confluent*.

In order to solve decision problems (i.e., decide languages), we use *families* of recogniser tissue P systems $\boldsymbol{\Pi} = \{\Pi_x : x \in \Sigma^\star\}$. Each input x is associated with a tissue P system Π_x that decides the membership of x in the language $L \subseteq \Sigma^\star$ by accepting or rejecting. The mapping $x \mapsto \Pi_x$ must be efficiently computable for inputs of any length, as discussed in detail in [5].

Definition 2. *A family of tissue P systems $\boldsymbol{\Pi} = \{\Pi_x : x \in \Sigma^\star\}$ is said to be (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:*

- *$F(1^n) = \Pi_n$, where n is the length of the input x and Π_n is a common tissue P system for all inputs of length n, with a distinguished input cell.*
- *$E(x) = w_x$, where w_x is a multiset encoding the specific input x.*
- *Finally, Π_x is simply Π_n with w_x added to its input cell.*

On the other hand, 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 $x \in \Sigma^\star$.

Any explicit encoding of Π_x is allowed as output of the construction, as long as the number of cells and objects represented by it does not exceed the length of the whole description, and the rules are listed one by one. This is also called a permissible encoding [5].

The class of problems solved by uniform (resp., semi-uniform) families of confluent tissue P systems with cell division is denoted by $\mathbf{PMC}_{\mathcal{TDC}}$ (resp., $\mathbf{PMC}^{\star}_{\mathcal{TDC}}$); the corresponding classes for tissue P systems with separation are $\mathbf{PMC}_{\mathcal{TSC}}$ and $\mathbf{PMC}^{\star}_{\mathcal{TSC}}$. The inclusions $\mathbf{PMC}_{\mathcal{TDC}} \subseteq \mathbf{PMC}^{\star}_{\mathcal{TDC}}$ and $\mathbf{PMC}_{\mathcal{TSC}} \subseteq \mathbf{PMC}^{\star}_{\mathcal{TSC}}$ hold by definition, since uniformity is a special case of semi-uniformity.

Finally, we recall the definitions of the complexity classes $\#\mathbf{P}$ and $\mathbf{P}^{\#\mathbf{P}}$ [7].

Definition 3. *The complexity class* $\#\mathbf{P}$ *consists of all the functions* $f\colon \Sigma^\star \to \mathbb{N}$, *also called* counting problems*, with the following property: there exists a polynomial time nondeterministic Turing machine* N *such that, for each* $x \in \Sigma^\star$*, the number of accepting computations of* N *on input* x *is exactly* $f(x)$.

Definition 4. *The complexity class* $\mathbf{P}^{\#\mathbf{P}}$ *consists of all decision problems solvable in polynomial time by deterministic Turing machines with oracles for* $\#\mathbf{P}$ *functions. These are Turing machines* M^f*, with* $f \in \#\mathbf{P}$*, having a distinguished* oracle tape *and a* query state *such that, when* M^f *enters the query state, the string* x *on the oracle tape is replaced in one step with the binary encoding of* $f(x)$.

3 Simulating tissue P systems

When simulating a tissue P system, we can limit ourselves to explicitly storing the configuration of the external environment (i.e., its multiset of objects), since this is where the result objects *yes* and *no* ultimately appear. This configuration can be stored in polynomial space by keeping track of the multiplicities of the objects in binary, with a special marker for those appearing with infinite multiplicity.

This is possible as long as we have a way to update this configuration even when not storing the configurations of the cells; this requires computing the multisets of objects communicated from or to the environment at each computation step. We are going to prove that such task can be performed in polynomial time by querying a $\#\mathbf{P}$ oracle, by adapting the proof of an analogous result for P systems with elementary active membranes [3].

The query we would ideally ask is "How much does the multiplicity of object a in the environment of Π change at time step t?"; however, we only know how to answer this query by simulating an entire computation of the tissue P system in polynomial space [11]. In order to try to reduce its complexity, we break it down into multiple queries with additional inputs describing the history of the computation up to the previous time step, and partially including the simulation of the current step. These extra inputs are computed using the answers to previous queries.

First of all, we need a way to distinguish multiple cells having the same label. Since cell division can create at most 2^t cells with the same label after t computation steps, we assume that each of these has a unique identifier in the range $[0, 2^t)$; we do not require the identifiers to be contiguous, or that a cell keep the same identifier during each step of the computation.

Since we are only dealing with confluent tissue P systems in this paper, we can also make assumptions on how the rules to be applied during each step must

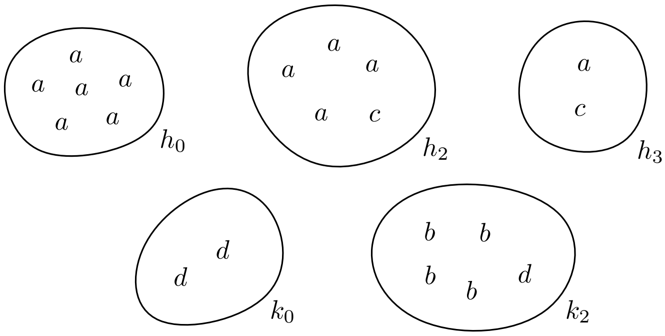

Fig. 1. Configuration of a tissue P system Π after two computation steps. The subscripts of the cell labels represent the identifiers of the corresponding cell.

be chosen. Without loss of generality, we give a *linear priority* to the rules, giving higher priority to communication rules, and applying a division (or separation) rule in a cell only when no communication occurs. Within the two groups of rules (communication versus division and separation), we fix an arbitrary total ordering. In particular, each communication rule is applied as many times as possible before applying any of those with lower priority.

We can now define a table associating each communication rule $[u]_h \leftrightarrow [v]_k$ with the set of identifiers of cells with labels h and k applying it at time t. Since describing arbitrary subsets of identifiers would require exponential space, we exploit once again the confluence assumption, and stipulate that each rule must be applied as many times as possible by all copies of h (resp., k) whose identifier belongs to a range of the form $[0, M_h)$ (resp., $[0, M_k)$) for some upper bound M_h (resp., M_k), where *zero* is an allowable number of applications. This corresponds to establishing another priority, over cells sharing the same label, given by the numerical ordering of the identifiers.

Definition 5. *A* communication table *for a tissue P system Π is a function $T\colon R \times \mathbb{N} \to \mathbb{N}^4$ such that, for $r = [u]_h \leftrightarrow [v]_k$ and $t \in \mathbb{N}$,*

$$T[r, t] = (M_h, \Delta_h, M_k, \Delta_k)$$

denotes that the cells with label h where rule r is applied at time t are those having identifiers in the range $[0, M_h]$; in particular, the rule is applied as many times as possible for identifiers strictly lower than M_h, and Δ_h times (a non-maximal number of times) for identifier M_h. The values M_k and Δ_k, symmetrically, denote the instances of cell k where r is applied.

A procedure for computing a communication table for a tissue P system is described later, as a portion of Algorithm 1.

Example 1. Consider the configuration in Fig. 1 of a tissue P system Π after two computation steps, with three instances of cell h (having identifiers 0, 2, and 3) and two instances of cell k (with identifiers 0 and 2), and consider the following two communication rules:

$$r_1 = [aa]_h \leftrightarrow [b]_k \qquad\qquad r_2 = [c]_h \leftrightarrow [d]_k$$

By giving priority to r_1 over r_2, and to lower identifiers over higher ones, we determine a unique way to apply the rules: rule r_1 is applied three times between h_0 and k_2, and once between h_2 and k_2, while rule r_2 is applied once between h_2 and k_0, and once between h_3 and k_0. Notice that k_0 applies r_1 zero times, which happens to be maximal in this case (since k_0 does not contain any copy of b). Thus, the smallest ranges of identifiers for h where r_1 and r_2 are applied *maximally* are $[0, 2)$ and $[0, 4)$, respectively, while those for k are $[0, 3)$ and $[0, 1)$, respectively. Furthermore, h_2 applies r_1 one extra time. Thus, according to the reasoning above, the communication table for Π has $T[r_1, 2] = (2, 1, 3, 0)$ and $T[r_2, 2] = (4, 0, 1, 0)$.

Notice that a communication table for the first t steps of Π can be stored in polynomial space with respect to t and the length of the description of Π.

Let us now focus on the simulation of tissue P systems with division only, and let us formulate a query that allows us to perform this task without simulating the individual cells.

Query Q. *Given a tissue P system with division $\Pi = (\Gamma, E, w_1, \ldots, w_d, R)$, a time step t in unary notation, a communication rule $r = [u]_h \leftrightarrow [v]_k$, and a communication table T for Π, with entries $T[\rho, \tau]$ filled for all $\tau < t$ and for $\tau = t$ if ρ has priority over r, how many times is rule r applied at time t by cells with label h, assuming the availability of enough copies of v in cells with label k?*

An oracle for query Q allows us to simulate tissue P systems with cell division with a polynomial slowdown.

Lemma 1. $\mathbf{PMC}^{\star}_{\mathcal{TDC}} \subseteq \mathbf{P}^{Q}$.

Proof. Let $L \in \mathbf{PMC}^{\star}_{\mathcal{TDC}}$ be a language, and let $\boldsymbol{\Pi} = \{\Pi_x : x \in \Sigma^\star\}$ be a semi-uniform family of tissue P systems with division deciding L in polynomial time. Algorithm 1 describes how each Π_x can be constructed and simulated, given the input string x, by a deterministic Turing machine with an oracle for Q.

In line 1 we obtain the description of Π_x by simulating the machine providing the semi-uniformity construction for $\boldsymbol{\Pi}$ on input x. This, by definition, can be carried out in polynomial time with respect to the length of x.

The loop of lines 2–14 is executed for each simulated time step t, hence, by hypothesis, a polynomial number of times. Inside this loop, the algorithm iterates across all communication rules $r = [u]_h \leftrightarrow [v]_k$ of Π in priority order (lines 3–9) in order to fill the corresponding entry $T[r, t]$ of the communication table.

We begin (line 4) by assuming that all existing copies of h, i.e., the full range of identifiers $[0, 2^t)$, are allowed to apply rule r, as if there were enough copies of multiset v among the copies of k; we make the same assumption for k. We then ask the oracle for Q how many times rule r is applied in cells with label h (line 6) and k (line 7) under those assumptions; call p and q those two numbers

```
1  construct $\Pi_x = (\Gamma, E, w_1, \dots, w_d, R)$ from $x$
2  for each time step $t$ do
3      for each rule $r = [u]_h \leftrightarrow [v]_k \in R$ in priority order do
4          $T[r, t] := (2^t, 0, 2^t, 0)$
5          repeat
6              $p :=$ no. of applications of $[u]_h \leftrightarrow [v]_k$ in $h$ at time $t$ according to $T$
7              $q :=$ no. of applications of $[u]_h \leftrightarrow [v]_k$ in $k$ at time $t$ according to $T$
8              update $T[r, t]$ by binary search
9          until $p = q$
10     for each rule $r = [u]_h \leftrightarrow [v]_0$ do
11         $p :=$ no. of applications of $r$ in $h$ at time $t$ according to $T$
12         remove $p$ instances of $v$ and add $p$ instances of $u$ to the environment
13     if yes or no appear in the environment then
14         accept or reject accordingly
```

Algorithm 1. Simulation of semi-uniform families of tissue P systems with cell division.

of applications. If $p \neq q$, then the number of copies of u in cells with label h differs from the number of copies of v in cells with label k; for the simulation to be consistent with the current configuration of Π, we need to ensure that $p = q$. Suppose, for the sake of example, that $p < q$. Then, we reduce the range of cells with label k by repeatedly adjusting the corresponding value M_k and re-evaluating q with further queries. By performing a binary search (line 8), we can find in polynomial time ($\log 2^t$ iterations) the smallest range $[0, M_k)$ of identifiers maximising the value of q, with the constraint $q \leq p$. The difference $p - q$ is finally recorded as Δ_k, the number of times r must be applied by the cell having label k and identifier M_k. (The argument is symmetric if the initial values of p and q are such that $p > q$.) This querying procedure is performed even if $h = 0$ or $k = 0$, i.e., one of them is the label of the environment.

The loop of lines 10–12 updates the configuration of the environment that we explicitly store, by asking the oracle the final number of applications of rules involving the environment, and adjusting the environment multiset accordingly. Notice that the rules not involving the environment are not simulated, since the configurations of the cells are not stored by Algorithm 1. In lines 13 and 14 the computation is halted when one of the result objects *yes* or *no* finally appears.

Since the number of queries needed, as well as the number of bookkeeping operations, is polynomially bounded, the simulation can be performed in $\mathbf{P}^Q$. □

In order to give a more precise upper bound of the complexity of simulating tissue P systems, we can now analyse query Q in detail, proving that it can be answered in polynomial time by a counting machine.

Lemma 2. *Query Q is in $\#\mathbf{P}$.*

Proof. Given a query Q with parameters Π, t, r, and T, Algorithm 2 describes a nondeterministic procedure for the parallel simulation of all cells of Π having

```
1  $id := 0$
2  for each time step $\tau \in \{0, \dots, t\}$ do
3      $newid := 2 \times id$
4      $newmultiset := \varnothing$
5      for each rule $\rho = [u]_h \leftrightarrow [v]_k$ in priority order do
6          $(M_h, \Delta_h, M_k, \Delta_k) := T[\rho, \tau]$
7          if $id < M_h$ then
8              remove as many copies of $u$ as possible from $multiset$
9              add the same number of copies of $v$ to $newmultiset$
10         else if $id = M_h$ then
11             remove $\Delta_h$ copies of $u$ from $multiset$
12             add the same number of copies of $v$ to $newmultiset$
13     if a rule $[a]_h \to [b_0]_h\ [b_1]_h$ is applicable then
14         nondeterministically guess a bit $i$
15         $newid := newid + i$
16         remove $a$ from $multiset$
17         add $b_i$ to $newmultiset$
18     $id := newid$
19     $multiset := multiset \cup newmultiset$
20 accept as many times as the no. of applications of $r$ in step $t$
```

Algorithm 2. Nondeterministic simulation of the cells having label h, with computation of the number of applications of communication rule r at time t.

label h, where each computation actually simulates a single cell. This algorithm manages the identifiers of the cells as follows: the identifier of the unique copy of cell h in the initial configuration is 0 (line 1); if the identifier of a copy of h at time τ is id, then in the next time step (line 18) the identifier is $2 \times id$ (line 3); if, furthermore, the cell divides, then the new copy, simulated by the computation where $i = 1$, has identifier $2 \times id + 1$ (line 15). This identifier schema is essentially identical to the one proposed by Sosík and Cienciala [11], and satisfies the two requirements described above: uniqueness among cells with the same label, and range $[0, 2^t)$ after t steps.

The algorithm simulates sequentially all steps up to t (line 2). In line 4 it initialises an empty multiset *newmultiset* to collect the objects entering the cell via communication rules, or rewritten via division rules; since the rules are simulated sequentially, we employ this auxiliary multiset (in addition to the actual content of the cell, named *multiset* in the pseudocode) in order to avoid applying more than one rule to each object.

The loop of lines 5–12 iterates across all communication rules ρ involving h (on either side of the rule). In line 6 we read the values corresponding to the ranges of identifiers for cell labels h and k where rule ρ is applied in the current time step. If the identifier of the cell being simulated belongs to the range $[0, M_h)$, then we apply rule ρ as many times as possible (lines 7–9). On the other hand, if the identifier is exactly M_h, we only apply the rule Δ_h times (line 10–12). The rule is not applied if the identifier is strictly greater than M_h.

If a division rule is applicable in the cell (this, in particular, requires that no communication rule was applied previously), then we apply the first one in priority order (line 13). This consists in nondeterministically choosing which of the two resulting cells the current computation will continue to simulate (line 14) and updating the identifier and contents of the selected cell (lines 15–17). Notice that this establishes a bijection between computations of the algorithm and instances of cell h.

We can then update the values of *id* and add to *multiset* the objects that appeared inside the current copy of cell h during the computation step just simulated (lines 18 and 19).

After having simulated t steps, we can check the number of times m that input rule r was applied in the cell during the last step. The algorithm can now "fork" m accepting computations[2] (line 20). This value contributes to the total number of accepting computations of the algorithm, which will then correspond to the number of applications of rule r at time t, as required. □

By combining Lemmata 1 and 2 we finally obtain our main result.

Theorem 1. $\mathbf{PMC}_{\mathcal{TDC}} \subseteq \mathbf{PMC}^{\star}_{\mathcal{TDC}} \subseteq \mathbf{P}^{\#\mathbf{P}}$. □

3.1 Tissue P systems with separation

Simulating separation rules $[a]_h \to [\Gamma_0]_h\ [\Gamma_1]_h$ instead of (or in addition to) division rules only requires a slight change to lines 13 and 17 of Algorithm 2. After having nondeterministically chosen which of the two resulting cells to simulate (bit i), we need to update *multiset* by removing the objects in Γ_{1-i}. Since this can also be performed in polynomial time, query Q remains in $\#\mathbf{P}$ and, as a consequence, the simulation of tissue P systems with separation has the same complexity.

Theorem 2. $\mathbf{PMC}_{\mathcal{TSC}} \subseteq \mathbf{PMC}^{\star}_{\mathcal{TSC}} \subseteq \mathbf{P}^{\#\mathbf{P}}$. □

4 Conclusions

We have proved a $\mathbf{P}^{\#\mathbf{P}}$ upper bound to the class of problems solvable in polynomial time by uniform or semi-uniform families of tissue P systems using division or separation rules. The simulation of tissue P systems we provided is also relatively robust with respect to the addition of features; for instance, it can be easily adapted to accommodate charges, evolution and dissolution rules from P systems with active membranes [3].

This is the same upper bound that holds [3] for P systems with active membranes where division can only be applied to elementary membranes (i.e.,

[2] This can be performed in polynomial time even if m is exponential, as it suffices to guess $\Theta(\log m)$ nondeterministic bits.

not containing further membranes). These two variants of P systems share the inability to create the complex nested structures of dividing membranes (such as exponentially large full binary trees) that allow unrestricted P systems with active membranes to solve **PSPACE**-complete problems in polynomial time [1]. It would be interesting to understand if it is possible to formalise this intuitive reasoning and link such membrane structure "complexity" with the ability of P systems to solve problems in polynomial time.

We do not know yet whether the $\mathbf{P}^{\#\mathbf{P}}$ upper bound is tight, or whether it can be lowered. Based on analogous results for P systems with active membranes [2], we conjecture that $\mathbf{P}^{\#\mathbf{P}}$ is indeed a precise characterisation of the problems solvable by general tissue P systems with division or separation; however, tissue P systems with maximum communication rule length (i.e., number of objects appearing in a communication rule) bounded by a small constant might prove to be weaker. It would be particularly interesting to analyse the borderline case of tissue P systems with division having rules of length at most 2, or those with separation having rules of length at most 3, which is the minimum necessary to solve classically intractable problems [8].

References

1. Alhazov, A., Martín-Vide, C., Pan, L.: Solving a PSPACE-complete problem by recognizing P systems with restricted active membranes. Fundamenta Informaticae 58(2), 67–77 (2003)
2. Leporati, A., Manzoni, L., Mauri, G., Porreca, A.E., Zandron, C.: Membrane division, oracles, and the counting hierarchy. Fundamenta Informaticae 138(1–2), 97–111 (2015)
3. 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 (2015)
4. Martín-Vide, C., Păun, Gh., Pazos, J., Rodríguez-Patón, A.: Tissue P systems. Theoretical Computer Science 296(2), 295–326 (2003)
5. Murphy, N., Woods, D.: The computational power of membrane systems under tight uniformity conditions. Natural Computing 10(1), 613–632 (2011)
6. Pan, L., Pérez-Jiménez, M.J.: Computational complexity of tissue-like P systems. Journal of Complexity 26(3), 296–315 (2010)
7. Papadimitriou, C.H.: Computational Complexity. Addison-Wesley (1993)
8. Pérez-Jiménez, M.J.: The P versus NP problem from the membrane computing view. European Review 22(01), 18–33 (2014)
9. Păun, Gh., Pérez-Jiménez, M.J., Riscos Núñez, A.: Tissue P systems with cell division. International Journal of Computers, Communications & Control 3(3), 295–303 (2008)
10. Sosík, P., Cienciala, L.: Computational power of cell separation in tissue P systems. Information Sciences 279, 805–815 (2014)
11. Sosík, P., Cienciala, L.: A limitation of cell division in tissue P systems by PSPACE. Journal of Computer and System Sciences 81(2), 473–484 (2015)