Research Statement

Mikaël Rabie

I first provide, in Sections 1 to 4, the works I focused during my thesis (some being pursued since my defense). I then describe my contributions made during internships I did before my PhD in Section 6. Since my PhD, I had two years of teaching contract, one in Paris and the other in Lyon. Section 6 describes work I initialized last year in the LAMSADE in the University Paris-Dauphine.

My main topics are Complexity, Computability, Distributed Systems and Graph Theory.

1 Related Works to my Thesis

Angluin et al. [3] proposed a model of distributed computation called population protocols. It can be seen as a minimal model that aims at modeling large sensor networks with resource-limited anonymous mobile agents. The mobility of the agents is assumed to be unpredictable (given by any fair scheduler) and pairs of agents can exchange state information when they are close together.

The seminal work of Angluin et al. [2, 3] proved that predicates computed by population protocols are precisely those on counts of agents definable by a first-order formula in Presburger arithmetic (equivalent to a semilinear set).

Angluin et al. added in [1] a time consideration to the model: Assuming that interactions are randomly chosen, what is the expected number of interactions to compute a predicate? Using several tools, they proved that with a unique leader in the input (an agent that is differentiated from all the others), any predicate computed by a classic population protocol can be computed in $O(n \log^5 n)$ expected interactions.

Guerraoui et al. introduced in [12] an extension of the model: The Community Protocols. Each agent has now a unique identifiers, and can store a fixed number of identifiers. With this tool, the model becomes way more powerful. It computes exactly what can be computed by non deterministic Turing Machines on space $O(n \log n)$. This result holds even with the presence of a finite number of byzantine agents in the population.

Chatzigiannakis et al., in [9], worked on an other variation: the Passively Mobile Communicative Machines. This time, each agent stores a Turing Machine tape of a finite size depending on the size of the population. Each agent has the same space of computation.

In this latter work, a hierarchy is provided according to the size $f(n)$ of the tape. With $f(n) = o(\log \log n)$, the model is as powerful as the classic population protocols. With
$f(n) = \Omega(\log n)$, any non-deterministic Turing Machine of tape of length $O(nf(n))$ can be simulated, as long as it is stable under input permutation.

I will now introduce the variations that I explored, in particular during my PhD.

2 Adding Rational Behaviour to Population Protocols

I worked on two approaches where we give to agents a rational behavior. We wanted to provide protocols that look ”intuitive” for some good definition, as we want agents to act rationally. The distributed dynamic can be then see as a social meeting. When two agents meet, they interact according to rules.

2.1 Pavlovian Population Protocols


In this section, we turn two players games into dynamics over agents, by considering PAVLOV behavior. The principle is simple: Agents are playing a game. Their input corresponds to their initial strategy. When two players meet, they ”play”. According to a payoff matrix, if their winnings is not enough, they switch their strategy to the best one against the strategy that just has defeated them.

In [6], we consider asymmetric games. We proved that any basic Presburger’s predicate can be simulated by a game. By adding the notion of multi-games, we prove that the model is equivalent to the original one.

As asymmetric games are not restrictive enough, in [5], we considered symmetric games, that appear to be more difficult to use. We provide games that checks if the population is of size greater than 3, or greater than $2^k$ for some $k$. We do not have any exact characterization, but I raised some conjectures.

2.2 Trustful Population Protocols

This section corresponds to some work published in [8] in DISC 13.

Here, we add a new restriction. If two agents meet and they both agree on a same opinion, they trust each other’s, and will not change their opinion. More formally, there is a map between possible outputs to possible opinions.

We provide an exact characterization of the model. It corresponds to the semilinear sets stable under multiplication and division (i.e. $x$ is in the computed set if and only if $\lambda x$ is in).

**Theorem 1** Sets computable by Trustful Population Protocols correspond exactly to the boolean combination of 0-threshold predicates of the form:

$$\left[ \sum a_i x_i \geq 0 \right]$$
To prove this, we provide a tighter bound when the possible outputs correspond exactly to the set of opinions. In this scenario, the set are stable under multiplication AND convex. This second restriction has not been characterized yet, but we do have a conjecture of what it is.

3 Weakening of the Community Protocols and Some Other Extensions

I focused on two modifications over the Community Protocols model [12]. I first looked what can be computed in $O(n \log^k n)$ expected interactions, for any $k$. I then focused on a weaker case, where identifiers can be shared by several agents.

3.1 Fast Computing with Community Protocols

The first restriction is actually a merge of two works. We know that any computable predicate by a Population Protocol can be computed in $O(n \log^6 n)$ expected interactions, as long as there is a unique leader at the beginning [1].

This gave the motivation to look to what can be computed in $O(n \log^k n)$ expected interactions (for any $k > 0$), with the Community Protocols model. We call this model $CPPL$. We can notice that $O(n \log^k n)$ expected interactions correspond to a polylogarithmic expected number of parallel interactions.

I proved that with this model, it is possible to encode in binary the size of the population, allowing to compute several languages and functions. I introduced several classes to compare it:

Theorem 2

\[ \text{POLYLOGTIME} \subseteq \text{CPPL} \subset \text{NSPACE}(n \log n) \cap \bigcup_{k \in \mathbb{N}} \text{SPACE}(n \log^k n) \]

I proved that the whole population cannot sort the identifiers in such a short time. The rational language $(ab)^*$ is not in $CPPL$.

I created a class of Turing Machine that is able to match everything we found to be computable. We did not find any exact characterization yet for the model. I published a short paper explaining my main results in [13] in SSS 16, and a longer version has been accepted in SIROCCO 17.

3.2 Homonym Population Protocols

Most of the results of this section have been published in [7] in NETYS 15, and a journal version has been accepted in the journal TOCS.

This time we followed a new restriction: what happens when a single identifier may be shared by several agents. More precisely, if we see identifiers as agent’s name, we accept
homonymy in the population. We then work through a hierarchy depending on the number $f(n)$ ($n$ being the size of the population) of identifiers present in the population. The hierarchy is summarized in the following table:

<table>
<thead>
<tr>
<th>$f(n)$ identifiers</th>
<th>Computational power</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(1)$</td>
<td>Presburger’s definable subsets</td>
</tr>
<tr>
<td>$\Theta(\log^r n)$ with $r \in \mathbb{R}_{&gt;0}$</td>
<td>$\bigcup_{k \in \mathbb{N}} \text{MNSPACE}(\log^k n)$</td>
</tr>
<tr>
<td>$\Theta(n^\epsilon)$ with $\epsilon &gt; 0$</td>
<td>$\text{MNSPACE}(n \log n)$</td>
</tr>
<tr>
<td>$n$</td>
<td>$\text{NSPACE}(n \log n)$</td>
</tr>
</tbody>
</table>

$\text{MNSPACE}(f(n))$ corresponds to non deterministic Turing Machines working on space $f(n)$ with an adapted input fit for this model.

For $f(n) \geq \log n$, I created an algorithm that writes the size of the population on $\log n$ agents, and then simulates a Turing Machine on the same space. By introducing the notion of tuples of identifiers, the main result for $\Theta(\log^r n)$ is found.

This result permitted to fill a gap in the hierarchy in [9], for the case where agents have Turing Machine with space $O(\log \log n)$. It corresponds exactly to $\bigcup_{k \in \mathbb{N}} \text{SNSPACE}(\log^k n)$.

With $o(\log n)$ identifiers, I found several non trivial algorithms, but not any characterization yet.

### 3.3 Population Protocols with other Tools

I considered new variation around the model of [12]. Identifiers are no longer ordered and only the equality test can be performed by agents.

Even if it looks to be too restrictive, it actually can still simulate a tape of size $O(n \log n)$. The only difference is that the input can only be seen as a multiset over the input alphabet, whereas in [12] the input can be a word, as it is possible to sort the elements of the input according to the order of the corresponding identifiers.

**Theorem 3** Without any order between identifiers, Community Protocols compute exactly $\text{SNSPACE}(n \log n)$.

I also proved that if agents have stack automata as tools, they can simulate any non deterministic Turing Machine.

### 4 Rusted Turing Machine

I introduced a new consideration of complexity for Turing Machines. I call a pivotal transition a step of a Turing Machine such as the machine’s internal state changes. I considered the
Machines such as the number of pivotal transitions does not depend on the input. It is equivalent to machines where the transition graph does not contain other cycles than loops. This later model will be called Rusted Turing Machine.

The case of machines with a unique internal state has been studied by Saouter in [14]. It corresponds to the machines with no pivotal transitions. This work helped these, as it proves the decidability of the halting problem.

Even if the model looks weak, I provided machines that computes:

- The size of the input written in binary.
- The semilinear sets.
- Any language of the form $u^*$.
- Some languages of the form $(u + v)^*$.
- The well-formed parenthesis language.

I proved in that the model cannot be compared to regular languages, context-free languages, and that star height of computable regular languages is not bounded.

The halting decidability provided in [14] permits to have the same result with this model. I found an upper bound of the number of cells reached with a single pivotal transition depending on the size of the input $x$ and the size of the alphabet $\Gamma$.

**Theorem 4** Before a single pivotal transition, a Turing Machine can write on at most $|\Gamma|^{|x|}$ extra cells.

From this, I obtain an upper bound of the number of cells reached with a Rusted Turing Machine, and hence an upper bound of the busy beaver of this model.

I did not get any explicit form of languages that cannot be computed by this model.

## 5  Internship works

Those two subsections are about two internships I followed in my first year of Master’s Degree. My other internships were focused on works that are described above, as they are related to my thesis.

### 5.1  Checkpointing Strategies

I worked on optimization use of multicores under the supervision of Frédéric Vivien and Yves Robert in the ENS Lyon. More precisely, how to duplicate jobs and schedule checkpoints in order to minimize the loss of time caused by processor failures. I worked on several distribution law of failures. I introduced the idea of job duplication, as using all the cores on a same job is dangerous with an expected time of failure exponentially increasing.

5.2 Trust in Game Theory

I looked at trust improvements for users, under the supervision of Shlomi Dolev in the Ben Gurion University (Israël). I introduced the idea to use trustworthy tools, like Coq, in order to help users believe in answers provided by external advisers. As an example, I implemented the notion of Nash Equilibrium in Coq, to show that it is possible to provide some equilibrium, and let a software check its veracity.


6 Maximisation de la Coordination

Those ongoing works have been started during my one year contract in the Paris-Dauphine University. It is a joint work with Jérôme Monnot, Florian Sikora, Bruno Escoffier and Sébastien Martin. It is a new approach on Graph Coloration. Each vertex has a set of admissible colors. We want to chose a color for each vertex maximizing the number of edges connecting two vertices with the same color. This is motivated by social coordination where people want to connect with the most of his contacts.

We proved that the problem is generally NP-complete. The NP-completeness holds even with the following constraints: 3 colors, bipartite planar graphs, 2 possible color per vertex, complete graph. The problem becomes polynomial with two colors, trees. We are currently working on approximations, fixed treedwith and color weight.

References


