Nothing can compare with a population, besides agents

Yann Busnel

To cite this version:

Yann Busnel. Nothing can compare with a population, besides agents. 2014. <hal-00933010>

HAL Id: hal-00933010

https://hal.archives-ouvertes.fr/hal-00933010

Submitted on 19 Jan 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Nothing can compare with a population, besides agents

Yann Busnel

LINA/Université de Nantes
2, rue de la Houssinière – BP 92208 – 44322 Nantes Cedex 03 – France
Fax: +33 (0)2 51 12 58 12 – Yann.Busnel@univ-nantes.fr

Abstract. Leveraging the resemblances between two areas explored so far independently enables to provide a theoretical framework for distributed systems where global behaviors emerge from a set of local interactions. The contribution of this paper arise from the observation that population protocols and multi-agent systems (MAS) bear many resemblances. Particularly, some subclasses of MAS seem to fit the same computational power than population protocols. Population protocols provide theoretical foundations for mobile tiny device networks. On the other hand, from long-standing research study in distributed artificial intelligence, MAS forms an interesting model for society and owns a broad spectrum of application field, from simple reactive system to social sciences. Linking the both model should offers several extremely interesting outcomes.

Keywords: Population Protocols, Multi-Agents System, Model Equivalence.

1 Introduction

With the recent expansion of distributed system the two last decades, a boundless set of models has been developed to capture diversity and richness of these systems. Among this set, some models, while proposed in different context, catch the same range of computability. Some recent studies deal with the comparison and equivalence between models [1,2,3,4,5]. In this paper, we investigate the connections between two famous models: Population Protocols and Multi-Agent Systems (MAS).

Population protocols [6] provide theoretical foundations for distributed systems in which global behavior emerges from a set of simple interactions between their agents. Originally developed in the context of mobile tiny devices, typically sensors, in this model, agents are considered anonymous, and therefore, undistinguishable. Many variants of population protocols have been proposed [7,8,9,10,11]. Among them, community protocols [12] augment the original model by assigning agents a unique identifier and letting nodes remember a limited number of other identifiers. This not only significantly increases the computation power of the system but also provides a way to tolerate a bounded

number of byzantine failures. In the sequel, the class of population protocols and variants will be referred as *population protocols*, and the original model as *basic population protocol*.

More specifically, the population protocol model consists in a finite space of agent's states, a finite set of inputs, a finite set of outputs and a transition function. The set of possible node's interactions is represented by a graph. When two agents are sufficiently close for a sufficiently long time, they interact by exchanging their local information, and update their state according to the transition function. For instance, if agents are small devices embedded on animals, an interaction takes place each time two animals are in the same radio range. The interaction patterns, orchestrated by a scheduler, are considered as unpredictable. Yet, the scheduler is assumed to be *fair i.e.*, it ensures that any reachable global system state can be reached infinitely often. In the absence of global knowledge, agents cannot usually verify that the protocol has terminated, therefore the model considers convergence (of the distributed output) rather than termination.

On the other hand, largely inspired by the social behavior analysis of insect colony, Multi-Agent System model aims at studying the concept of collective and/or distributed intelligence [13]. The former model and its several extensions should be viewed as a crossroads in artificial intelligence (AI), distributed AI, distributed systems, software engineering, and smart objects.

As proposed in [13] and enhanced in [14], a MAS is made up of an environment, wherein a set of active objects, so called agents, interact with passive objects. A set of relation links objects between them, according to their activity, as a set of operations defines what is possible to apply to objects. Finally, some operators are in charge to reflect the operation mapping on the environment (usually denoted "universe laws"). It exists a specific case where the environment is empty and its does not exists passive object. This kind of system is called *purely communicating MAS*, as the set of relations defines a network between agents, for which actions are reduced to communication. This specific structure is commonly used in distributed AI, mostly for collaboration between units design for problem solving of expert systems.

Usually, MAS are studied owing to their noteworthy ability for self-organization, and are sometimes surprisingly able to reproduce certain form of complex social systems. MAS could be split in two main classes: Cognitive and Reactive agents. These classes come from the design of the active agents, according to their algorithmic ability. More precisely, the cognitive class is commonly used for distributed AI as the reactive one permits to study virtual life [15]. In this paper, we mostly consider the latter one that shares several similarities with the Population Protocol model. Recent interest in the probabilistic convergence of MAS raises [16,17]. A motivation of our work relies on the fact that the latter convergence has been extensively studied in the Population Protocol model [7,8], and results should be simply bring into some MAS models.

As in [2] in which authors bridge the gap between population protocols and gossip-based protocols, the aim of this paper is then to correlate the two afore-

mentioned model: Population Protocols and Multi (reactive) Agents Systems, and more specifically, the ones made up of *eco-agents*. They both rely on finite-state agents. Both aim at achieving an emerging global behavior from a set of local interactions in a fully decentralized manner. The main contribution of this paper is to acknowledge these similarities and leverage them in both contexts.

This paper is organized as follow. In Section 2, we present a classification of Multi-Agent Systems. Second, Section 3 provides some background on basic population and community protocols. Then, we prove their equivalence in Section 4 and discuss some opportunities of leveraging it, before concluding in Section 5.

2 Different Kinds of Agent

The *Kenetic* [13] claims to be the science and technics of artificial organizations. Using MAS as a designing tool, this field of research possesses the global control of all the experimental system parameters as an advantage. Yet, if the designer is able to tune any possible parameters, she has no access to any determinism aspect, regarding the system evolution. Quite the contrary, as changing configuration in MAS are subject to chaotic phenomena. This implies that any variation in initial parameter, or any adding of insignificant random variable, should lead the same system to exactly opposite final state. In fact, these tiny changes are dramatically magnified through interactions between agents, avoiding some precise system state prediction.

All the Kenetic problems are located at a meeting point of the notion of agent and of society. In other words, it is at a crossroads in the relationship between individual behavior and globally observed phenomena. In this context, cooperation, conflict, collaboration and coordination of actions make sense. Let us restrain the possible field by defining what we mean by *agent* and multi-agent system.

2.1 MAS Model

MAS are not yet subject to any law. Any work in this model is located in the center of a duality: Agent and Organization. Any organization comes from agent's interactions, but the local behavior of these agents should be modified in return from the constraints imposed by the created organization structures. Except predefined organizations by designers, task coordination, assignment and distribution are resulted from agents themselves. Some emerging properties appear, without any initial programming. It is necessary to introduce two schools of thought according to the model appliance, mainly relying on architectural differences; these differences concern the algorithmic nature of used agents. One school of thought is specialized in distributed AI, so called *cognitive school*, as the other one, denoted *reactive school*, is more interested in virtual life. The one considered in this paper corresponds to this reactive one.

Cognitive agents Cognitive MAS design aims to reach communication and cooperation in classical expert systems. In this case, the MAS will be made up of a small number of "intelligent" agents. Each one owns a basic knowledge that includes a set of mandatory information and technical know-how in order to succeed in its task, and to manage interactions with its environment and other agents. They are sometimes called *intended*, *i.e.,* they own some goals, and explicit plans permitting to reach them. In the context of cognition, mostly planned, MAS leads to some group of individuals, govern by predefined social rules (for instance, in case of conflict, agents should be forced to negotiate). Researches in this field often rely on sociological studies about organization and small groups. In this paper, we do not consider further this kind of MAS.

Reactive Agents The main class considered in this paper relies on the fact that a globally intelligent system should not required individual intelligence of its agents. Simple reaction mechanism to stimuli should resolve complex problem, without short-term plan, or explanation of aims and objectives. These MAS permit to raise some self-organization, as the famous example of ant's colony. In the latter, as all ants are undistinguishable, and without any global authority, coordination raised from agents' action in order to develop and save the colony. This collective entity is able to resolve complex issues as construct the anthill, find food, take care of eggs and larva, *etc.* [18]. Emerging organization is principally related to reactive MAS, as they are not characterized by predefined structures (which is seldom, if ever, the case of cognitive MAS).

2.2 Formalism

Multi-Agent System We propose to model a MAS as proposed in [13].

Definition 1. *A* Multi-Agent System *(MAS) is a system made up of:*

- *An environment E,* i.e., *generally a metric space;*
- *A set of objects O. These objects are usually located,* i.e., *at each time, it is possible for any object to associate it with a position in E. Most of these objects are* passive, i.e., *they can be sensed, created, removed or modified by agents;*
- *A set of agents A such that $A \subseteq O$, which represent* active *entities of the system;*
- *A set of relations R that links objects (and thus agents) between them.*
- *A set of operations Op that lets agents in A to perceive, produce, consume transform and manipulate objects in O.*
- *Operators responsible for applying these operations and the world reaction to this modification. We denote these reactions* Universe Laws.

It exists a specific case of MAS where $A = O$ (*i.e.,* there is no passive objects) and where $E = \emptyset$. This kind of system is call *purely communicating MAS*, as the set of relations R defines a network between agents, for which actions are reduced

to communication. This specific structure is commonly used in distributed AI, mostly for collaboration between units and designed for problem solving of expert system. This organization emulates the modus operandi of a social structure as administration for instance.

Computational power of specific MAS relies on the potential of the environment [19]. The environment gives structure to the MAS, by providing a kind of shared memory, or a physical/spatial structure. In this study, we mainly focus on virtual environment using discrete representation. The environment could also abstract the communication structure, which is composed by an infrastructure for message passing, stigmergy or implicit communication. In cognitive MAS, it should also provide some social structure, defining roles, groups and communities of agents. We introduced then the notion of *autonomous environment*. An autonomous environment is defined as an environment with extended functionality compared with a simple spatial structure. The simplest way to model the environment is to provide only a metric space in order to estimate the ability of communication of any two agents, according to their distance between their respective locations. Any enhanced environment will be denoted as autonomous in the following.

Agent The core entity of MAS is the agent itself. Let us introduce a generic definition:

Definition 2. *An* agent *is a virtual or physical entity that:*

- *is able to act in an environment;*
- *is able to communicate directly toward other agents;*
- *is directed by a set of trends (as individual objectives or satisfaction function, or even survival);*
- *owns proper resources;*
- *is able to perceive (in a limited manner) its environment;*
- *owns a partial representation of this environment (possibly empty);*
- *owns abilities and should offer services;*
- *can be able to reproduce;*
- *has a behavior striving for satisfying its aims, by tacking into account available resources and skills, according to its perceptions, representations and communications.*

As introduced in Section 2.1, agents should be cognitive or reactive. However, this splitting is sometimes to simplistic. We need to enhance it according to two axes, which are summarized in Table 1:

- Agent's comportments are usually leaded by their teleonomic behavior[1] in contrast with the reflex infers by perception. The trends that govern agents should come either from the environment, or explicitly expressed in agents. We mention it as a *reflex* behavior in the first case, and a *teleonomic* one in the second one.

[1] "*Teleonomy*" represents the scientific concept of end in itself (purposefulness and goal-directedness of structures and functions in living organisms).

Behavior \ Environment relation	Cognitive agents	Reactive Agents
Teleonomic	Intentional Agents	Instinctual Agents
Reflex	"Modules" Agents	Tropic Agents

Table 1. Characterization of agents

– Agent's relation with the environment raises the classical problem of subject/object. Is an agent have an explicit and symbolic representation of the surrounding world, on which she is able to reason or this representation is sub-symbolic, *i.e.,* integrated in its sensorimotor intelligence. In the first case, we mention *cognitive* agent, and in the second, *reactive* agents.

2.3 Eco Problem Solving

The **Eco Problem Solving (EPS)** is a decentralized approach of problem solving, using emergence of interacting reactive agents [20]. A stable state is then sought and considered as the solution of the problem.

Any eco-agent has only one purpose: to be satisfied. Its environment is composed by the other neighboring agents. The latter agents should be in an acquaintance or in a dependency network, interacting using perception of actions. Two specific perceptions of eco-agent exist: being aggressed and being hampered. Three different actions can then be chosen: reach satisfaction (carrying the action that reach its objective), aggress another agent, or take flight. It can be represented as a finite state automaton, with four-intern state: Satisfied (S), Seeking satisfaction (SS – usually the entry state), Seeking to take flight (ST) and Taking flight (TF). A transition relation is then set between these states.

The actions made by an agent in its acquaintance or its dependency network follow the rules hereafter:

– if an agent is satisfied, it informs his dependencies that they should be satisfied also;
– if an agent cannot be satisfied yet, it seeks the hampering agents among its acquaintance and aggresses them;
– if an agent seeks to take flight, it seeks the hampering agents among its acquaintance and aggresses them.

EPS can be applied in very different context [20], from task scheduling (using "task agents" and "resources agents"), to production line optimization, including simulation of emergent structural turbulence stability in fluid dynamics. Depending on the applicative requirement, eco-agent should own an identifier or not. The latter case infers a specific class of EPS so-called *Anonymous EPS* in the rest of this paper.

However, the scope of possibilities of eco-agent is not restricted to problem solving. It is especially adapted to evolving universe simulation. In fact, in an

eco-agent system, an external stimuli is treated as any other entry of the problem. Any agent acts locally in response of a perturbation. It is perfectly illustrated by the *Misachieving baby* problem [21]. It introduces some malicious external adversaries, which are very powerful and raise the difficulty of eco-resolution.

3 About Population Protocols

In this section, we briefly present the basic population protocol model and the community protocol variant, which relaxes the assumption on the anonymity of agents.

3.1 Basic population protocol

The basic population protocol model, initially introduced in [6], is composed of a collection of agents, interacting pairwise in an order determined by a fair scheduler. Each agent has an input value and is represented by a finite state machine. This agent can only update its state through an interaction. Updates are defined by a transition function that describes the function f computed by the system. At each interaction, the agents compute an output value from their current state and converge eventually to the correct output value, depending to the inputs initially spread to the agents.

More formally, a population protocol is composed of:

- a complete interaction graph Λ linking a set of $n \geq 2$ agents;
- a finite input alphabet Σ;
- a finite output alphabet Y;
- a finite set of possible agent's states Q;
- an input function $\iota : \Sigma \to Q$ mapping inputs to states;
- an output function $\omega : Q \to Y$ mapping states to outputs;
- a transition relation $\delta : Q \times Q \to Q \times Q$ on pairs of states.

In the sequel, we call $(p, q) \mapsto (p', q')$ or (p, q, p', q') a transition if $(p, q, p', q') \in \delta$. A transition can occur between two agents' states only if these two agents have an interaction. The protocol is deterministic if δ is a function (*i.e.,* at most one possible transition for each pair in Q^2).

A configuration of the system corresponds to an unordered multi-set, containing states of all agents. We denote $C \to C'$ the fact that a configuration C' can be obtained from C in one step (*i.e.,* with only one transition for one existing interaction). An execution of the protocol is a finite or infinite sequence of population configurations $C_0, C_1, C_2, \ldots$ such that $\forall i, C_i \to C_{i+1}$.

As introduced above, the order of the interactions is unpredictable, and is decided by the scheduler. The scheduler is assumed to be *fair*, *i.e* a feasible configuration cannot be endlessly ignored. In other words, if a configuration C appears an infinite number of times during an execution, and there exists a possible step $C \to C'$, then C' must also appear an infinite number of times in the execution. This ensures that any attainable configuration is eventually reached.

3.2 Community protocols

Many variants of the last model exist. In this paper, we focus on the specific extension so-called community protocol [12], which significantly increases the computational power of the basic population protocol. This model augments the basic population protocol model by assigning unique identifiers to agents. All possible identifiers and a special symbol $\perp$ are grouped in an infinite set U. The difference between basic population protocols and community protocols is the definition of the set of states: $Q = B \times U^d$ where B is the initial definition of the population protocol's set of states collapsed to a memory of d identifiers. As in population protocols, algorithms cannot use any bound on the number of agents and moreover, U is infinite. In order to maintain the population protocol spirit in this extended model, some constraints are added: only existing agent identifiers can be stored in the d slots intended for identifiers of an agent's state and no other structural information about identifiers can be used by algorithms. We consider, for $q \in Q$ and $id \in U$, that $id \in q$ means that q stores id in one of its d identifier slots. Thus, community protocols have to verify the two following formal constraints:

$$\forall (q_1, q_2, q_1', q_2') \in \delta, id \in q_1' \vee id \in q_2' \Rightarrow id \in q_1 \vee id \in q_2. \tag{1}$$

Consider π a permutation of U with $\pi(\perp) = \perp$. For $q = \langle b, u_1, u_2, \ldots, u_d \rangle \in Q$, let $\hat{\pi}(q) = \langle b, \pi(u_1), \pi(u_2), \ldots, \pi(u_d) \rangle$ We assume that:

$$\forall (q_1, q_2, q_1', q_2') \in \delta : (\hat{\pi}(q_1), \hat{\pi}(q_2), \hat{\pi}(q_1'), \hat{\pi}(q_2')) \in \delta. \tag{2}$$

In short, the first assumption ensures that no transition introduce new identifiers and the second one that identifiers can only be stored or compared for equality, but not manipulated in any other way. Any population protocol can be viewed as a community protocol with $d = 0$.

Finally, a population or community protocol stably computes a function $f : \Sigma^+ \to Y$ if $\forall n \in \mathbb{N}, \forall \sigma \in \Sigma^n$, every fair execution with n agents initialized with the elements of σ, eventually stabilizes to output $f(\sigma)$. That means that the output value of every agent eventually stabilizes to $f(\sigma)$.

3.3 Computational power

In order to characterize computable functions of population protocols, Angluin *et al.* study a set of decidable predicate restriction. In fact, the *basic population protocol* model characterization is precisely identical than the set of *semi-linear predicate* [6,22]. A *set of semi-linear* is a subset of $\mathbb{N}^d$ made up of finite union of *linear set* such that $\{ \overrightarrow{b} + k_1\overrightarrow{a_1} + k_2\overrightarrow{a_2} + \cdots + k_m\overrightarrow{a_m} \}$, where $\overrightarrow{b}$ is the d-dimension reference vector, $\overrightarrow{a_1}$ to $\overrightarrow{a_m}$ are basis vectors, and k_1 to k_m are non-negative coefficients. A *semi-linear predicate* on an entry set corresponds to a predicate that is validated exactly on a semi-linear set.

In [6], Angluin *et al.* prove that basic population protocols are able to compute any predicate define using *Presburger's arithmetic*, which precisely match

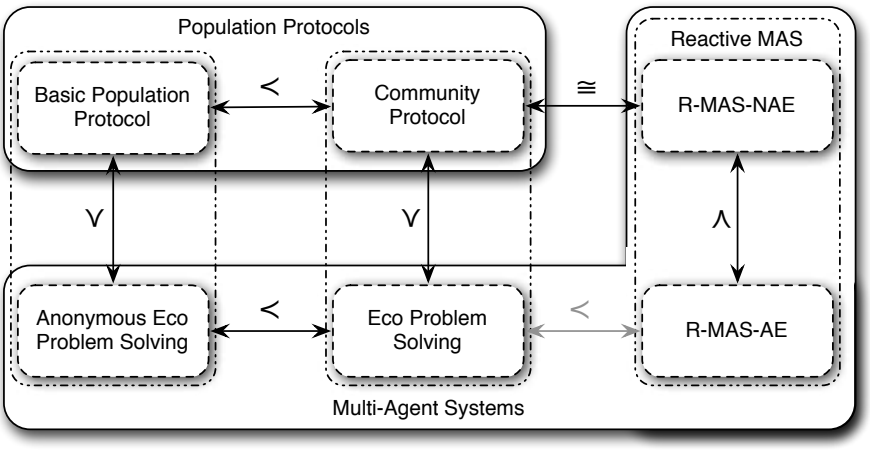

Fig. 1. Relationship between Multi-Agent Systems and Population Protocols

with the set of all the semi-linear predicates. Presburger's arithmetic is a first-order theory of the natural number, in the Peano's arithmetic language without the multiplication operation (*i.e.,* with only the addition operation, eventually ordered, besides zero and successor operator). Moreover, any semi-linear predicate should be compute using a population protocol [22].

Guerraoui and Ruppert provide a proof that their extended model drastically increases the latter power [12]. Indeed, CPs are equivalent to the class of the symmetric functions in $NSPACE(n \log n)$; *i.e.,* any language decided using a non-deterministic Turing machine using $O(S \log S)$ slots, with respect to the following condition. Consider a function $f : \Sigma^+ \to Y$. f is *symmetric* if, for all string y obtained by swapping character from another string x, we have $f(x) = f(y)$. In other words, $\forall x \in \Sigma^+, \forall \pi$ permutation of x,

$$f(x) = f(\pi(x)) \Leftrightarrow f \text{ is symmetric.}$$

4 Population *vs.* Reactive Agents

Recently, the probabilistic convergence of MAS has been studied apart [16,17]. A motivation of our work relies on the fact that the latter convergence has yet been extensively studied in the Population Protocol model [7,8], and these results should be simply bring from one model to another.

In this section, we prove the classification and the relations between all the considered models that are summarized in Figure 1 where R-MAS-AE (resp. R-MAS-NAE) means Reactive MAS with Autonomous Environment (resp. without Autonomous Environment). This provides a refined classification of MAS and population protocols. Let us split the proof in three main theorems.

First of all, the two left horizontal relations of Figure 1 are straightforward by extension of [2], as they are based on the fact that increasing the model with unique agent identifier allows to extend the computational power of the former model. Thus, we obtain that Anonymous-EPS $\prec$ EPS and PP $\prec$ CP. On the other hand, the rightest vertical relation between R-MASs is also straightforward as an autonomous environment can obviously simulated a non-autonomous environment, using a restriction on the environment of the R-MAS-AE. The reverse side is impossible as agents in R-MAS-NAE are not able to deterministically compute the unique evolution of their environment. Thus, we trivially obtain that R-MAS-NAE $\prec$ R-MAS-AE. In the following sections, we provide proofs of remaining relations illustrated on Figure 1.

4.1 Anonymous EPS weaker than Basic Population Protocol

We show there that there exists a population protocol $\mathcal{P}$ able to simulate any execution of eco-resolution.

Theorem 1. *If f is computable by a protocol from Anonymous EPS, then there exists a basic population protocol which can compute f.*

Proof. Let $\mathcal{E}$ an EPS using eco-agents that computes a specific function f. As presented above, a finite-state machine models eco-agents.

Mapping the domain of the transition function: The domain of any relation in R or operation in Op is finite (and corresponds to some Cartesian products of O, the set of objects, with $\mathcal{D}_S$ the set of possible states of an object in the environment). Moreover, as elements in $R \cup Op$ are functions, theirs ranges are also finite by definition. Based on these sets, we define $\mathcal{D}_{\mathcal{E}}(g)$ a specific subset of the Cartesian product between the domain and the codomain of $g \in R \cup Op$ (*i.e.,* $\mathcal{D}_{\mathcal{E}}(g)$ contains each ordered pair such that the first entry is in the domain of g and the second entry is the mapped element of this first entry by g). Thus, for any $g \in R \cup Op$, $\mathcal{D}_{\mathcal{E}}(g)$ is finite and contains all the possible transitions of object states, based on the knowledge of a surrounding object sub-state in the environment.

Design of the basic population protocol for the purpose of simulation: Consider the following basic population protocol $\mathcal{P}$, represented by the 7-uplet $(\Lambda, \Sigma, Y, Q, \iota, \omega, \delta)$. Consider a complete interaction graph Λ. Let the set of agent states in $\mathcal{P}$ be identical to the set of object states, *i.e.,* $Q = \mathcal{D}_S$. Consider that Σ and Y are the same as the input and output sets of $\mathcal{E}$, if they exist. In this case, ι and ω are the same functions than the ones which respectively associate the input set of $\mathcal{E}$ to $\mathcal{D}_S$, and $\mathcal{D}_S$ to the output set of $\mathcal{E}$. Conversely, if no specific input and output sets are defined in $\mathcal{E}$, then $\Sigma = Y = \mathcal{D}_S$ and $\iota \equiv \omega$ corresponds to the identity function. Finally, the transition function δ is defined as follows.

$$\forall g \in R \cup Op, \forall (s_l, s_r, s'_l) \in \mathcal{D}_{\mathcal{E}}(g), \exists (s_r, s_l, s'_r) \in \mathcal{D}_{\mathcal{E}}(g) \text{ s.t. } (s_l, s_r, s'_l, s'_r) \in \delta.$$

On the environment and the fair scheduler: Communication exchange between agents is an inherent characteristic of EPS. However, in this class, the

environment is only used to model the ability for two agents, or an agent and an object, to pairwise interact. Therefore, as the scheduler of $\mathcal{P}$ is mandatory fair, any possible interaction eventually happens and makes sufficient condition to obtain a correct execution of $\mathcal{E}$, using the aforementioned $\mathcal{P}$.

Thus, there exists a basic population protocol $\mathcal{P}$, which simulates the considered EPS $\mathcal{E}$ and computes the function f. Then, for any function computable by an EPS, there exists a basic population protocol, which stably computes this function. □

On the other hand, the reverse of Theorem 1 is not valid. In fact, there exists some functions that are computable by a population protocol, but no anonymous EPS is able to compute the same function. This is mainly due to the global fairness property associated to the Population Protocol model. In some EPS, it is possible to construct a loop-based infinite execution that endlessly avoid a given reachable system state, belying the aforementioned fairness property.

4.2 EPS weaker than Community Protocol

Along the same lines, we prove here the following theorem in order to prove that it is possible to simulate any EPS using a community protocol.

Theorem 2. *If f is computable by a protocol from EPS, then there exists a community protocol which can compute f.*

Proof. Let $\mathcal{E}$ an EPS using eco-agents that computes a specific function f. Thus, each eco-agent in the system is aware of its unique identifier. Consider the following community protocol $\mathcal{C}$.

Preliminary assumptions on $\mathcal{C}$: We assume that, in the community protocol used on $\mathcal{C}$, agents are uniquely identified, and a unique agent is assigned a specific identifier $id_{\mathcal{L}}$. This agent is considered as the leader by all other agents. In this specific community protocol, we assume that this leader is aware of the size of the system[2] (denoted n in the sequel).

Design of the community protocol for the purpose of simulation: The only difference between a basic population protocol and a community protocol consists in the definition of the set of states (*i.e.,* $Q = B \times U^d$) and the two constraints on the state's identifier part (*i.e.,* the part belonging to U^d cannot be used freely). Similarly, the only difference between Anonymous-EPS and EPS remains on the use of identifiers.

Then, due to the proof of Theorem 1, the community protocol $\mathcal{C}$ has to be designed to simulate a given EPS $\mathcal{E}$. Then, we can still consider the fair scheduler as the environment. In order to do that, the state of an agent in $\mathcal{C}$ is the same than for the eco-resolution one, enriched by its own coordinate in the environment.

[2] This assumption is only expected for the synchronization barrier. It can be relaxed using the probabilistic clock phase mechanism proposed for population protocols in [7].

Dealing with the environment simulation in $\mathcal{C}$: For each interaction, δ is applied on involved agents if and only if these two agents are in adjacent coordinates. Moreover, some specific state required surrounding environment knowledge. For instance, to simulate the Taking Flight state, the corresponding agent must check if there is any space in front of it. In order to verify such property, $\mathcal{C}$ uses a synchronization barrier to check it as follows.

Consider an agent id that fall into ST state. In this state, it only waits until it encounters the agent $id_{\mathcal{L}}$. During its next interaction with $id_{\mathcal{L}}$, id sets its state to *wait* and $id_{\mathcal{L}}$ store the current coordinates of id in its own state. For all futher interaction between $id_{\mathcal{L}}$ and others agents, $id_{\mathcal{L}}$ store the position of this object/agent and the latter also sets its state to *wait*. Thus, all agents eventually stabilize to the *wait* state, and the number of coordinated stored on $id_{\mathcal{L}}$ eventually converges to n (the number of agents in the population). At this point, all agents have reached the synchronization barrier.

After the barrier, $id_{\mathcal{L}}$ enables all agents to resume their own execution by release their *wait* state as described in [2], and gradually remove stored coordinates after these interactions. The set of these stored coordinated eventually becomes empty and all agents eventually leave the *wait* state of the last barrier, and are ready for their next action. In the meanwhile, $id_{\mathcal{L}}$ waits to first interact with id before starting this releasing phase. This ensure that id will be aware of the position of all the objects in the environment and is then able to check if there is an empty space in front of it.

In conclusion, if $\mathcal{E}$ computes the function f, then the community protocol $\mathcal{C}$ simulates the behavior of $\mathcal{E}$ and also computes the function f. □

Again, the opposite of Theorem 2 cannot be verified, due to the same argument about to the fairness property (see Section 4.1).

4.3 R-MAS-NAE is equivalent to Community Protocol

Finally, we prove here the following theorem in order to prove the equivalence between community protocols and R-MAS-NAE.

Theorem 3. *A predicate is computable by a community protocol if and only if it can be computed by a Reactive MAS without Autonomous Environment (R-MAS-NAE).*

Proof. The proof of Theorem 3 is directly inferred from the statements of Lemmata 1 and 2, which show respectively both implications of this equivalence. □

Consider the first implication of this theorem. Based on the usage of the environment as the scheduler, the following lemma is almost straightforward.

Lemma 1. *For each f computable by a community protocol, there exists a R-MAS-NAE which compute f.*

Proof. Let $\mathcal{C}$ the community protocol computing f and defined by the 7-tuples $(\Lambda, \Sigma, Y, Q, \iota, \omega, \delta)$. Consider the R-MAS-NAE $\mathcal{M}$ described below. We have to show that $\mathcal{M}$ simulates $\mathcal{C}$. First of all, each agent in Λ is hosted by a specific agent of $\mathcal{M}$. Input, output and state sets are the same in $\mathcal{M}$ than in $\mathcal{C}$. *A fortiori*, both map function ι and ω remain identical in $\mathcal{M}$.

Dealing with the transition function: The $\mathcal{M}$'s relations R and operations Op are defined from δ: the universe laws is only composed of the possible transition extracted from δ. Moreover, we consider a purely communication MAS (see Section 2.2) where R is restricted to pairwise communication (*i.e.,* each time agents interact with others, they could only operate a pairwise transformation state that exists in δ). Thus, the sequence of population configurations is valid and represents the community protocol $\mathcal{C}$, as it only stems from the transition function δ using pairwise interactions.

On the fairness assumption: The last assumption to verify is the fairness condition. In R-MAS-NAE, the scheduler is fully defined by the (non-autonomous) environment evolution, which, as we mentioned before, potentially leads to any possible communication according to the current distance between each agent. In that context, every possible finite scheduling has a non-null probability to happen. Thus, every possible transitions between two system configurations $C \to C'$ has a non-null probability to happen. The fairness assumption is then verified.

We are then able to conclude that $\mathcal{M}$ simulates the community protocol $\mathcal{C}$, which computes the function f. □

Finally, let now show the opposite of Lemma 1, corresponding to the second part of Theorem 3.

Lemma 2. *For each f computable by a R-MAS-NAE, there exists a community protocol which computes f.*

Proof. Let $\mathcal{M}$ the given R-MAS-NAE and consider the following community protocol $\mathcal{C}$.

Summary of agent state requirement: Due to space constraints, we do not present the formal mapping between possible agent states and the transition function, which are trivially extended from the proof of Theorem 2.

Dealing with the environment simulation in $\mathcal{C}$: Consider a specific agent in $\mathcal{C}$ that store the whole state of the environment of $\mathcal{M}$, which has a finish state (as the set of object is finished). Following the outline of the *rendez-vous* mechanism presented in the proof of Theorem 2 (and in more details in [2]) with the *environment agent* as the leader, it becomes simple to simulated $\mathcal{M}$ with $\mathcal{C}$. Indeed, as the environment is not autonomous, there is no change in the latter one without direct interaction with an object of $\mathcal{M}$. Thus, each action of any agent of $\mathcal{M}$ should infer a synchronization barrier, an update of the environment (on the aforementioned dedicated agent), and finally, a global release that lets the other agent opportunity to also interact with the environment.

In conclusion, if $\mathcal{M}$ computes f, then the community protocol $\mathcal{C}$ simulates the behavior of $\mathcal{M}$ and also computes the function f. □

We then have proved all the claims presented in Figure 1.

5 Conclusion and future works

The main contribution of this paper is to establish a correlation between Population Protocols and Multi (reactive) Agents Systems. Both aim at achieving an emerging global behavior from a set of local interactions in a fully decentralized manner. This parallel between two worlds, explored so far independently, offers several extremely interesting outcomes, acknowledge these similarities and leverage them in both contexts. These results can be leveraged for existing results as well as results to come in both areas. For instance, the intensive research around structured language for MAS (as KQML[23] or FIPA ACL[24] for instance) should be mapped to the population theory. That should lead to a standardization of high-level communication between agents. From the inverse, the theoretical analysis of Population Protocols models could resolve the issue of a lack of computational power analysis with different MAS models.

Acknowledgment

The author would like to warmly thank Philippe Lamarre (INSA Lyon, France) for his valuable discussions at the early steps of this work.

References

1. Basu, A., Bonakdarpour, B., Bozga, M., Sifakis, J.: Systematic correct construction of self-stabilizing systems: a case study. In: 12th international conference on Stabilization, safety, and security of distributed systems (SSS '10), Springer-Verlag (2010) 4–18
2. Bertier, M., Busnel, Y., Kermarrec, A.M.: On gossip and populations. In: the 16th International Colloquium on Structural Information and Communication Complexity (SIROCCO 2009), Piran, Slovenia (2009)
3. Marzougui, B., Hassine, K., Barkaoui, K.: A new formalism for modeling a multi agent systems: Agent petri nets. J. Software Engineering Applications **12** (2010) 1118–1124
4. Perathoner, S., Wandeler, E., Thiele, L., Hamann, A., Schliecker, S., Henia, R., Racu, R., Ernst, R., Harbour, M.G.: Influence of different system abstractions on the performance analysis of distributed real-time systems. In: 7th ACM/IEEE international conference on Embedded software. EMSOFT '07, ACM (2007) 193–202
5. Poslad, S.: Specifying protocols for multi-agent systems interaction. ACM Transactions on Autonomous and Adaptive Systems **2** (2007)
6. Angluin, D., Aspnes, J., Diamadi, Z., Fischer, M.J., Peralta, R.: Computation in networks of passively mobile finite-state sensors. Distributed Computing (Special Issue: PODC'04) **18** (2006) 235–253
7. Angluin, D., Aspnes, J., Eisenstat, D.: Fast computation by population protocols with a leader. Distributed Computing (Special Issue: DISC'07) **21** (2008) 183–199

8. Angluin, D., Aspnes, J., Eisenstat, D., Ruppert, E.: The computational power of population protocols. Distributed Computing (Special Issue: DISC'07) **20** (2007) 279–304
9. Angluin, D., Aspnes, J., Fischer, M.J., Jiang, H.: Self-stabilizing population protocols. In: 9th International Conference Principles of Distributed Systems (OPODIS '05), Pisa, Italy (2005) 103–117
10. Aspnes, J., Ruppert, E.: An introduction to population protocols. Bulletin of the European Association for Theoretical Computer Science, Distributed Computing Column **93** (2007) 98–117
11. Delporte-Gallet, C., Fauconnier, H., Guerraoui, R., Ruppert, E.: When birds die: Making population protocols fault-tolerant. In: 2nd IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '06), San Francisco, CA, USA (2006) 51–66
12. Guerraoui, R., Ruppert, E.: Names trump malice: Tiny mobile agents can tolerate byzantine failures. In: 36th International Colloquium on Automata, Languages and Programming (ICALP '09). Volume LNCS 5556. (2009) 484–495
13. Ferber, J.: Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence. Addison Wesley, London, UK (1999)
14. Sycara, K.P.: The many faces of agents. AI Magazine (1998) 11–12
15. Gechter, F., Chevrier, V., Charpillet, F.: A reactive agent-based problem-solving model: Application to localization and tracking. ACM Transaction Autonomous and Adaptative Systems **1** (2006) 189–222
16. Martinez, S.: A convergence result for multiagent systems subject to noise. In: American Control Conference, 2009. ACC '09. (2009) 4537–4542
17. Tang, G., Guo, L.: Convergence of a class of multi-agent systems in probabilistic framework. In: Decision and Control, 2007 46th IEEE Conference on. (2007) 318–323
18. Drogoul, A., Corbara, B., Lal, S.: Manta: New experimental results on the emergence of (artificial) ant societies. In Gilbert, N., Conte, R., eds.: Artificial Societies: the computer simulation of social life. UCL Press, London, UK (1995)
19. Weyns, D., Omicini, A., Odell, J.: Environment as a first class abstraction in multiagent systems. Autonomous Agents and Multi-Agent Systems **14** (2007) 5–30
20. Drogoul, A., Dubreuil, C.: Eco-problem-solving model: results of the n-puzzle. ACM SIGOIS Bull. **13** (1992) 15–
21. Schoppers, M.: Universal plans for reactive robots in unpredictable environments. (1987) 1039–1046
22. Angluin, D., Aspnes, J., Eisenstat, D.: Stably computable predicates are semilinear. In: 25th annual ACM symposium on Principles of distributed computing (PODC '06), Denver, CO, USA (2006) 292–299
23. Finin, T., Fritzson, R., McKay, D., McEntire, R.: Kqml as an agent communication language. In: Proceedings of the third international conference on Information and knowledge management, ACM (1994) 456–463
24. Fipa, A.: Fipa acl message structure specification. Foundation for Intelligent Physical Agents (2002)