Formalization, Implementation, and Modeling of Institutional Controllers for Distributed Robotic Systems

José N. Pereira[1], Porfírio Silva[2], Pedro U. Lima[3], Alcherio Martinoli[4]

[1,4] Distributed Intelligent Systems and Algorithms Laboratory, EPFL, Lausanne, Switzerland
[1,2,3] Institute for Systems and Robotics, Instituto Superior Técnico, Lisbon, Portugal

[1,4] {jose.pereira | alcherio.martinoli}@epfl.ch, [2,3] {porfiriosilva | pal}@isr.ist.utl.pt

[1,4] EPFL ENAC IIE DISAL, Station 2, CH-1015 Lausanne, Switzerland

[2,3] Instituto de Sistemas e Robótica, Instituto Superior Técnico, Torre Norte - 6.18, Av. Rovisco Pais 1, 1049-001 Lisboa, Portugal

[1] +41 21 693 4797, [2] +351 21 841 8196, [3] +351 21 841 8274, [4] +41 21 693 6891

[1] Corresponding Author

Keywords
Distributed Robotic Systems, Institutional Robotics, Executable Petri Nets, Robotic Controllers, Stochastic Modeling

Abstract

The work described in this paper is part of a long term goal of introducing Institutional Robotics, a novel framework for the coordination of robot teams that stems from Institutional Economics concepts. Under the framework, institutions are cumulative sets of persistent artificial modifications made to the environment or to the internal mechanisms of a subset of agents, thought to be functional to the collective order. In this paper we introduce a formal model of institutional controllers based on Petri nets. We define Executable Petri Nets, an extension of Petri Nets that takes into account robot actions and sensing, to design, program and execute institutional controllers. We use a generalized stochastic Petri net view of the robot team controlled by the institutional controllers to model and analyze the stochastic performance of the resulting distributed robotic system. The ability of our formalism to replicate results obtained using other approaches is assessed through realistic simulations of up to 40 e-puck robots. In particular, we model a robot swarm and its institutional controller with the goal of maintaining wireless connectivity, and successfully compare our model predictions and simulation results with previously reported results, obtained by using finite state automata models and controllers.

1 Introduction

Despite recent advances in the study of cooperation (within robots and with humans) in multi-robot systems, the social environments considered are still rudimentary. Differentiation between interactions as part of a larger set of social rules or as interactions with the environment is fundamental in obtaining complex cooperative behavior, in as much the same way as the differentiation of a hundred dollar bill from a piece of an equivalent colored paper is fundamental in economic relations. Several concepts from social sciences have been applied to multi-agent systems (e.g., norms [10], trust [18], reputation [9]) in order to understand how different social interactions shape the behavior of the whole system. We believe that such a variety of approaches is generally too fragmentary and that an unifying concept is needed to give extra consistency to the study of social artificial intelligent systems.

Institutional Robotics (IR) [19] is an approach to cooperative distributed robotic systems, that draws inspiration from the social sciences, namely from Institutional Economics' concepts [11]. It combines the notions of institution, coordination artifact, and environment, aiming to provide a comprehensive strategy for specifying social interactions (e.g., norms, roles, hierarchies) among robots. Under IR, robots are situated not only in a physical but also in an institutional environment, where their interactions are guided by institutions. Through cooperative decision-making, these institutions can be modified by the robots, providing adaptation to a changing scenario. Cooperation is achieved by this regulation of social interactions since the robots know not only how to behave in a given scenario but also what to expect from other robots and the environment.

One example of an institution is a roundabout. Roundabouts have a *material* side, since they act as obstacles to the regular straight-line motion of vehicles on a road. However, they further have a *mental* side, specified by the rule that states that a roundabout must be navigated on a counter-clockwise direction (in most countries, but differently from the UK and Australia). While a roundabout specifies a behavior for a single vehicle, the social interactions that occur when multiple vehicles navigate a roundabout in opposite directions result in a coordinated effort, avoiding collisions between vehicles.

One of the goals of our research is to formalize the concepts of IR from a computer science perspective, so as to create an ontology of the entities which are part of the IR model, and to represent formally their connections, as well as the corresponding coordination and control algorithms. A further goal of our research is to develop IR models that predict the system performance both quantitatively and qualitatively, and to analyze its intrinsic limitations, performance bounds, and general system properties (e.g., liveness).

In this paper, we first focus on formalizing the central concept of IR - institutions. Institutions were defined in [20] as cumulative sets of persistent artificial modifications made to the environment or to the internal mechanisms of a subset of agents, thought to be functional to the collective order. Institutions

can be viewed as coordination artifacts encapsulating relevant behavioral rules for robots (possibly designed based on problem-domain knowledge), that specify social interactions of different types among actors in a given scenario (robots, humans, software agents, etc.). Each institution is modular, representing one desired behavioral rule, and allowing the construction of a complex robot behavioral controller by the composition of several institutions. They represent the basic building blocks for creating cooperative working environments for social robots and humans. Modularity allows us to distinguish social interactions and to differentiate them from interactions with the environment, contributing to a richer social environment for multi-robot systems and their interactions with humans.

We formalize institutions using an abstract representation, allowing their design and execution for distributed robotic systems, so as to obtain behaviors capturing the social interactions of interest. The use of an abstract representation combined with the intended modularity enables institutions to be shared by robots and to coordinate the collective behavior. For instance, if an institution is modified during execution, either by cooperative decision-making or by designer intervention, this change can be propagated to the entire robotic team without the need of stopping execution to re-implement the controllers. In order to accomplish this objective we propose to use Petri Nets as an abstract representation for institutions. Our method composes a set of institutions, to create an institutional robot controller able to execute a desired task and observe the specified social interactions.

We also assess in the paper the ability of our IR formalism to replicate results obtained with other approaches. We consider a case study concerned with a robot swarm which has to maintain wireless connectivity, design a set of institutions to perform this task, and obtain an institutional controller. We successfully compare results from realistic simulations performed using the institutional controller to those performed with the original finite state automata controller [22].

Finally, we show that a Generalized Stochastic Petri Net (GSPN) view of institutional controllers in closed loop with the distributed robotic system they control, enables qualitative and quantitative model-based analysis. We use the GSPN view of the institutions designed for our case study to derive a macroscopic model that captures the mean-field dynamics of the distributed robotic system. We compare our model predictions with results obtained in realistic simulation.

In Section 2 we discuss related work and background for our formalization. This formalization is presented in Section 3 culminating with the definition of a controller based on our institutional approach. In Section 4 we apply this formalism to the wireless connected swarm case study. Section 5 presents our modeling methodology and model results.

2 Background and Related Work

Institutional economics is a fundamentally different approach from neo-classical theory, the current trend of economics and inspiration for market-based systems of task allocation in distributed robotics [8].

In [11], Hodgson refines a description of institutional economics outlining the following main features: institutions are the key element of any economy; the economy is an open and evolving system; and the notion of individuals as utility-maximizing agents is inadequate. The institutional approach is characterized also by the rejection of unbounded rationality. Agents are affected by the institutional environment they live in, but in no way does that environment fully determine their behavior. Every agent has individual goals and motivations that it wants to fulfill. Institutions are developed by these very same agents.

IR [19] aims to provide a comprehensive strategy for specifying social interactions among robots, by combining the notions of institution, coordination artifact, and environment. According to the IR approach:

1. the coordination strategy is supported by a network of institutions;
2. institutions are coordination artifacts of different types (e.g., norms, roles, hierarchies);
3. robots are able to modify both their physical and their institutional environment;
4. robots need a high degree of autonomy, pursuing goals based on their "struggle for survival".

From an institutional perspective, institutions are taken as the main tool of any sophisticated society, and individuals are both constructive within and constructed through institutional environments. In a first attempt at formalizing institutions in the IR model [20], Silva et al. define them as "cumulative sets of persistent artificial modifications made to the environment or to the internal mechanisms of a subset of agents, thought to be functional to the collective order".

This definition is too abstract to be applied "as is" to distributed robotics experiments. Thus, we go back to the idea of institutions as coordination artifacts [21]. Coordination artifacts [16; 17] are infrastructure abstractions in multi-agent systems meant to improve the synthesis and analysis of coordination activities. The main properties that describe coordination artifacts are: *specialization*, *encapsulation*, and *inspectability*. Specialization refers to the fact that coordination artifacts are specialized in automating coordination activities and can be represented with concurrency frameworks such as Petri Nets or process algebras. Coordination artifacts encapsulate a coordination service, allowing the agents to abstract how it is implemented. Encapsulation is the key to achieve reuse of coordination. Inspectability refers to the property that an artifact should support some procedure to allow engineers or agents responsible for the system to check for errors in its specification.

Omicini et al. [16] argue that coordination artifacts are exterior to the agents using them and perceived as individual entities, but can actually be distributed on several nodes of a multi-agent system. We propose that, when taking institutions as coordination artifacts, they can be part of the agent controller, working as norms or procedures the agent has to follow. Even with this assumption, we can still think of institutions being distributed in our multi-robot system, if we consider their representation to be replicated in each agent.

In our research we are interested in borrowing some concepts from a multi-level probabilistic modeling methodology established for swarm robotic systems, that takes into account the individual robot controller in order to generate an aggregated macroscopic representation of the dynamics of the whole team. This methodology was proposed by Martinoli et al. [12] for a collaborative swarm robotics case study and has been applied to many other case studies including two directly relevant for this paper and concerned with aggregation [4] and wireless connectivity [22].

3 Petri Nets and Institutions

Starting from the concept of institutions as coordination artifacts we model them using a formal representation, leading to a standard design and execution platform (in real robots, realistic simulations, and multi-agent systems). Institutions encapsulate relevant behavioral rules for robots, specifying social interactions of different types among actors in a given scenario. They represent the basic building blocks for creating shared coordinated working environments. Moreover, concurrent execution of institutions has to be regulated since not all behaviors can be executed simultaneously. Considering the three main properties of coordination artifacts mentioned above, we propose to use Petri Nets (PN) as the formal framework.

Our choice of PN is based primarily on the ability of this formalism to deal with distributed systems. State information is distributed among a set of places that capture key conditions that govern the operation of the system. Moreover, PNs can represent an adequate computational model for effective and efficient interaction management, a key aspect of coordination artifacts. In our method, different sets of places will govern the execution of actions available to the robot and the state of execution (active or idle) of each of the behaviors encapsulated by an institution.

PNs are also well suited to deal with modularity. A multi-layer methodology, introduced in [5], enables one to organize hierarchically the interaction between multiple behaviors, and their PN-based implementations. While interactions are described at a higher layer, the implementation of each behavior can be described at a lower layer in a modular fashion. This property of PNs is fundamental for our intended modular representation of institutions. Petri Net Plans [23] is another example of tool specifically directed to the design and execution of robotic plans using Petri Nets. PNs are also well suited for dealing with a variable

degree of coordination in the distributed system, including scenarios in which a tight inter-node coordination via dedicated, explicit communication channels is required. From this perspective, PNs are better suited than Finite State Automata (FSA) approaches to deal with the concurrency of tightly coordinated systems, a required condition for our intended modular approach to institutions. Moreover, using FSA, representing different combinations of what institutions are active might lead to an intractable increase in the number of states. In the PN representation, it is possible to keep a moderate growth of the graph size, compared with the growth in number of states.

3.1 Executable Petri Nets

We follow the definitions for Petri Nets and their dynamics (enabled transitions, state transition dynamics) in [3]:

Definition: A *Petri Net* is a five-tuple (P, T, A, w, X) where:

- P is the finite set of *places*;
- T is the finite set of *transitions*;
- $A \subseteq (P \times T) \cup (T \times P)$ is the set of arcs from places to transitions and from transitions to places;
- $w : A \to \mathbb{N}^+$ is the *weight function* on the arcs;
- X is a *marking* of the set of places P, $X = [x(p_1), \ldots, x(p_n)] \in \mathbb{N}^n$ represents the *state* of the Petri Net.

Herein, we assume that all the weights of the arcs are 1. If $x(p_i)$ in marking X is equal or larger than 1, we say that place p_i is marked. Each unit in $x(p_i)$ is called a *token*, i.e., if $x(p_i) = 1$ then p_i has one token. State transitions in Petri Nets occur by moving tokens through the net and changing the marking by doing so. The sets of input places $I(t_j)$ and output places $O(t_j)$ of a transition t_j are given by $I(t_j) = \{p_i \in P : (p_i, t_j) \in A\}$ and $O(t_j) = \{p_i \in P : (t_j, p_i) \in A\}$. Petri Net dynamics are provided by the following state transition function:

Definition: The *state transition function*, $f : \mathbb{N}^n \times T \to \mathbb{N}^n$, of Petri Net (P, T, A, w, X) is defined for transition t_j if and only if

$$x(p_i) \geq w(p_i, t_j) \quad \text{for all } p_i \in I(t_j) \tag{1}$$

If $f(X, t_j)$ is defined, then we set $X' = f(X, t_j)$, where

$$x'(p_i) = x(p_i) - w(p_i, t_j) + w(t_j, p_i), \quad i = 1, \ldots, n \tag{2}$$

If transition t_j verifies condition (1) then we say it is *enabled*. When transition t_j is enabled, we say that it can *fire*, and thus trigger a state change on the net by moving tokens according to (2).

Our aim is to formalize institutions for modeling and execution of robot controllers, using PNs. This means that we need to take into account robot actions and sensor readings. We consider three sets of building blocks that will allow us to design our controllers:

- the set Act contains all robot primitive actions (combinations of two or more primitive actions are not considered as primitive actions);
- the set Cdt contains boolean conditions that can be verified by checking sensor readings;
- the set Pac contains "parameter actions", which are auxiliary actions not concerning actuators but that only modify variables needed for the actions in Act.

Thus, the Petri Nets used for execution of robot and institution behaviors are defined as follows.

Definition: An *Executable Petri Net* (EPN) is a Petri Net (P, T, A, w, X) where:

- each place $p_i \in P$ has an associated action $a_i \in Act$;
- each transition $t_i \in T$ has an associated condition $c_i \in Cdt$ and an associated parameter action $pa_i \in Pac$.

The basic intuition behind this definition is that by associating actions with places we are able to define which actions are to be executed at each time step. This is done simply by checking if the corresponding place is marked. By associating transitions with conditions verified by sensor readings we trigger state changes in the Petri Net due to changes in the robots environment. Algorithm 1 (Fig. 1-(a)) is performed by the robots at each time step, allowing the robots to execute the behavior modeled by an EPN.

The implementation code for actions and conditions in the sets Act, Cdt and Pac is not explicitly represented in the specification of an EPN. All robots share a common function table that implements all possible actions and conditions. These are then represented in the EPN by means of indices. This allows the EPNs to be generic, in a sense that although robots may have different implementations for the same action (e.g., heterogenous robots in terms of hardware), the same EPN could be used to achieve coordination in the same manner. Also, it enables the sharing of EPNs among robots without the sharing of the actual implementation of actions.

3.2 Institutional Agent Controller

Our goal is to formalize institutions as coordination artifacts in a modular fashion. We represent each institution by an EPN that can be executed independently or together with other institutions. We also represent robot's *individual behaviors* by EPNs. While the institutions specify behaviors that have a *social nature*, i.e., they relate the robot to other robots in some way, the individual behaviors specify a set of basic behaviors that have exclusively an *individual nature*, i.e., they relate the robot with the surrounding environment and its own goals. The composition of the individual behavior with a set of institutions generates a robot controller.

The distinction between institution and individual behavior is a design choice, based in the differentiation between social interactions and interactions with the environment. Individual behaviors specify basic behaviors, closely related to the robots own goals and their "struggle for survival" and initially set by the designer. Even if individual behaviors produce some interactions with other robots, these are not planned and result from the dynamic nature of the environment. On the other hand, institutions specify socially constructed behaviors that further help robots achieve their goals and coordinate with the rest of the team. Interactions through institutions are distinguished as social interactions and the robots may deliberate how to change or how to conform to them.

A previous abstract definition of institution was presented in [20]. There, the authors define the institution as a tuple *(ID, Rationale, Modifiers, Network, Institutional Building, History)*, where each element of the tuple tries to capture the main constitutive elements of the social order dynamics. For our purpose of formalizing institutions using an abstract representation, allowing for a standard design and execution platform, this definition is not sufficient.

The composition of the individual behavior with a set of institutions is non-trivial since concurrent execution of some of the institutions might be impossible or at least inadequate to the task the robot is carrying out. An example of such institutional interplay is that an institution stating that you must drive on the right side of the road will be overruled by the institution of the road code of Great Britain, and thus should not be executed when in that territory. In [7], Crawford and Ostrom define a set of deontic operators, $D = \{P, O, F\}$, establishing permitted (P), obliged (O), and forbidden (F) operations, to be applied to institutional statements in order to deal with this problem. In our formalization, these operators affect whether institutions are active or idle at each time step. However, the conditions that govern when a specific institution is active might refer directly to the activity state of other institutions. For instance, the institution for driving on the right is forbidden (and thus should be idle) when the institution of the road code of Great Britain is active. This referencing of other institutions creates a problem for our intended

modular approach to formalization. Therefore, we have chosen to use a more restrictive set of deontic operators in order to guarantee that institutions do not refer to any other specific institution but can still prevent the concurrent execution of undesired behaviors (individual robot behaviors and other institutions in general).

Definition: The set D of deontic operators for IR institutions includes the following deontic operators: $\{AllowAll, StopInd, StopInst, StopAll\}$. Their corresponding definitions are as follows:

- *AllowAll* implies that the associated institution can be executed concurrently with the individual behavior and all the other institutions;
- *StopInd* implies that the associated institution cannot be executed concurrently with the individual behavior;
- *StopInst* implies that the associated institution cannot be executed concurrently with other institutions;
- *StopAll* implies that the associated institution cannot be executed concurrently with the individual behavior or other institutions.

We now present our formalized definition of institution:

Definition: An *Institution* I is a four-tuple $(Inst, initial_I, final_I, d_I)$ where:

- $Inst$ is an EPN;
- $initial_I$, $final_I \in Cdt$ are initial and final conditions for the execution of $Inst$;
- $d_I \in D$ is the associated deontic operator.

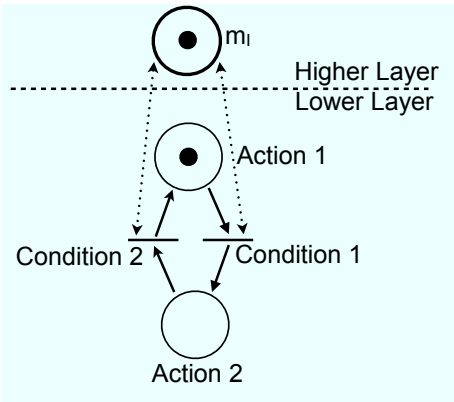
The EPN $Inst$ specifies the desired behavior that should be performed by the robot. This behavior is not always being executed, its start and end are dictated by conditions $initial_I$ and $final_I$, which the robot verifies at each time step. Thus, we say that an institution I at each time step can be *active* or *idle*. Each institution also includes a deontic operator d_I which is used when combining it with the robot individual behavior and further institutions. $Inst$ must be designed, but institutions can be kept simple and further behavioral complexity is the result of composition, in a modular fashion.

As previously mentioned, Petri Nets (and thus EPN) can be represented by macro places in a hierarchical fashion, using two distinct layers. We consider that individual behaviors and institutions are part of a lower layer and are represented by one macro place in the higher layer, as shown in Fig. 1-(b). In the lower layer the EPN $Inst$ of institution I is displayed. In the higher layer the macro place m_I representing institution

Algorithm 1 Execute Petri Net

```
repeat
    for all enabled transitions t_i ∈ T do
        if associated condition c_i is true then
            run associated parameter action pa_i
            fire transition t_i
        end if
    end for
until no transition has fired
for all marked places p_i ∈ P do
    run associated action a_i
end for
```

(a)



(b)

Figure 1: (a) *Execute Petri Net* algorithm. (b) Hierarchical representation of an EPN in two layers. Dotted arcs represent two directional arcs, one from a transition to a place and one from a place to a transition. Bottom: lower layer, EPN $Inst$ with conditions and actions associated to transitions and places. Top: higher layer, macro place m_I.

I is displayed. By adding arcs from each transition in $Inst$ to m_I and from m_I to each transition (shown as a single bidirectional dotted arc), we guarantee that each transition will only be enabled if m_I is marked. When a transition in $Inst$ fires, m_I will continue to be marked since it is a output place of the transition.

Thus, if a macro place is marked, the individual behavior or institution that it represents is active, otherwise it is idle. This allows us to compose our institutions at the higher layer where relationships among the institutions and the individual behavior should be specified while keeping relationships between actions and conditions separated in the lower layer. Both layers can be then merged algorithmically to obtain a full EPN that can be used as controller.

The composition of individual behaviors and institutions is controlled by the deontic operators as presented in Fig. 2-(a). As stated before, composition takes places only in the higher layer. We will see how different deontic operators for institution I_1 control the composition while always maintaining the deontic operator of institution I_2 as $AllowAll$. If the deontic operator of institution I_1 is also $AllowAll$, then no other relationship is necessary since all behaviors can be executed concurrently and the net is presented in Fig. 2-(a), without the places and transitions encapsulated by the dashed line boxes.

If the deontic operator of I_1 is $StopInd$, the structure encapsulated by the top dashed line box is added. Place $idle_{Ind,I1}$ represents the individual behavior being idle because of institution I_1 being active. The added transitions have associated a special condition that is always true. This specifies that if institution I_1 is activated, then the individual behavior is set to idle and vice-versa. If the deontic operator of I_1 is

StopInst, the structure encapsulated by the bottom dashed line box is added. This means that institution I_2 can be idle if place $idle_{I2}$ is marked or if place $idle_{I2,I1}$ is marked. On the latter case, institution I_2 will resume being active when institution I_1 becomes idle. Our setup considers only two institutions but the structure would be added for all institutions except I_1, if more institutions were present.

If the deontic operator is *StopAll* then we consider a combination of the previous two cases, with structures encapsulated by both dashed line boxes being added. These rules also apply for institution I_2 if it has a different deontic operator than *AllowAll*. The procedure for composition of individual behavior and institutions (by adding the control structures discussed above to their higher layer representation) is algorithmic and has been automated.

We can now define an Institutional Agent Controller:

Definition: An *Institutional Agent Controller* (IAC) is an EPN resulting from the composition of an individual behavior Ind and a set of institutions $\{I_1, \ldots, I_n\}$ controlled by the deontic operators $d_{I_1}, \ldots, d_{I_n}$.

Each robot in a social collective setting mediated by institutions runs its IAC. All macro places and control places ($idle_i$) added during composition are associated with a void action. Considering these associations, our IAC is itself an EPN and can be executed by Algorithm 1 (see Fig. 1-(a)). A minor change is needed to line 9 of the algorithm to make sure that not only the lower layer place is marked but also the higher layer macro place of the institution being executed.

The overall effort needed for the formalization includes the design time of both institutions and individual behavior as well as their composition. While the latter is performed algorithmically with negligible investment, the former requires a certain amount of time and experience with design of behavior-based controllers. The IAC for a desired task can be obtained prior to an experiment and uploaded to the robots. It is also possible for each robot to obtain the IAC from a given set of institutions at the start of the experiment. Thus, the method is fully scalable to any number of robots. The complexity of the IAC increases only with the number of institutions.

4 Wireless Connected Swarm Case study

In this section we present a case study to illustrate how to apply our formalism in order to obtain an IAC that performs the desired task. Our aim is to be able to specify behaviors that have a social nature as institutions and behaviors that have an individual nature as individual behaviors. We compare results from simulations with different controllers to assess the ability of our IAC approach to replicate results obtained with other

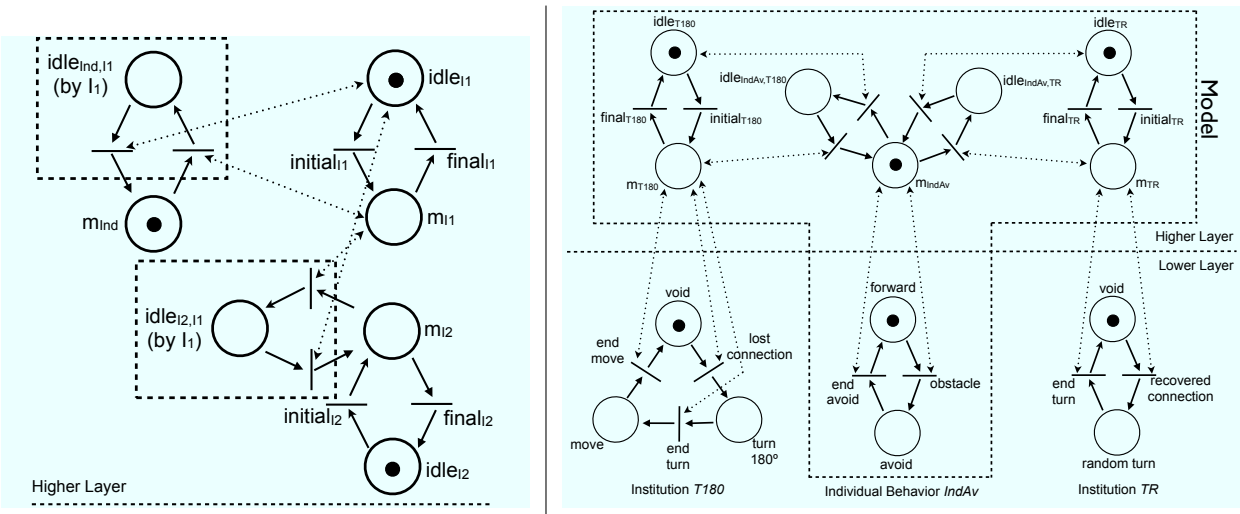
approaches.

We have selected a case study previously investigated in [15; 22], where a decentralized control algorithm is able to maintain a certain degree of spatial compactness of a robotic swarm (with N robots) in an unbounded arena using exclusively, as information at the robot level, the current number of wireless connections to the neighbors. The communication is local and its bounded range a parameter of the robotic system. Let X be the number of connections perceived by a robot. In the default state (defined as $forward$), the robot simply moves forward. If at any time the robot senses the loss of a connection and X falls below a threshold α (where $\alpha \in \{0, \ldots, N-1\}$), the robot assumes it is going in the wrong direction and switches to state *coherence*. In this state the robot performs a 180° turn in order to recover the lost connection. Upon recovering the lost connection, the robot performs a random turn and moves back to the default state. If the connection is not recovered, the robot simply moves to the default state. If an obstacle is detected the robot immediately switches to state *avoid*, where it performs obstacle avoidance for a given number of time steps, after which it returns to its previous state.

While this simple algorithm has limited robustness, it allows the swarm to maintain its connectivity to a certain extent, with its spatial compactness being controlled by the communication range and by the threshold α. It is implemented in [22] using a FSA with states defined as above.

In our IAC implementation, robots execute an individual behavior $IndAv$ (*Individual Avoidance*) and two institutions $T180$ (*Turn 180 degrees*) and TR (*Turn Random*), all specified by EPNs shown in the lower layer of Fig. 2-(b). Individual behavior $IndAv$ specifies a behavior relating the robot to its environment. The robot moves forward and, upon detection of an obstacle, performs obstacle avoidance for a given number of time steps, after which it returns to moving forward. The robot updates its neighborhood information (sends and receives messages) both while moving forward and avoiding obstacles. We do not consider this to be part of a behavior with a social nature since in the individual behavior robots do not act upon information provided by other robots. Institutions $T180$ and TR implement the social rules, dealing with loss and recovery of connections. $T180$ specifies that upon losing a connection the robot performs a 180° turn followed by moving forward for a small number of steps. This short forward movement allows the robot to assess if the lost connection was recovered immediately after the execution of $T180$. Institution TR specifies that if a connection is recovered the robot performs a random degree turn.

To consider institutions as defined in Section 3, we need initial and final conditions and deontic operators. For institution $T180$ we say that initial condition $initial_{T180}$ is "loss of connection detected and number of connections is less than α" and the final condition $final_{T180}$ is "move forward procedure has ended". For institution TR we say that initial condition $initial_{TR}$ is "recovery of connection detected and previous number of connections is less than α" and the final condition $final_{TR}$ is "random turn procedure has ended".



(a) (b)

Figure 2: (a) Higher layer composition scheme for two institutions I_1, I_2 and individual behavior Ind. Dotted arcs represent bidirectional arcs, as in Fig. 1. Places m_{I1}, m_{I2}, m_{Ind} are higher layer macro places representing implementations of institutions and the individual behavior in the lower layer. These representations will be used throughout the paper. Places and transitions encapsulated by dashed line boxes are added during the composition procedure. (b) IAC for wireless connected swarm. Lower layer: EPNs for individual behavior $IndAv$ and institutions $T180$ and TR. Higher layer: composition of individual behavior and institutions. PN structure for GSPN model encapsulated in dashed line box.

The deontic operator associated with both institutions is $StopInd$, specifying that institutions and individual behavior cannot be executed concurrently.

We now have all the elements needed to obtain the IAC that specifies our desired behavior. The composition of the individual behavior $IndAv$ and institutions $T180$ and TR (specified separately by EPNs shown in the lower layer of Fig. 2-(b)) is shown in the higher layer of Fig. 2-(b). The final controller is the full EPN of Fig. 2-(b), obtained after merging the two layers. Both composition and merger are performed algorithmically without the need of further designer input. Lower layer actions and conditions are implemented in the robot. Thus, to perform the task the robot needs only to execute Algorithm 1 taking the IAC as input. Actions associated with marked places are executed, much in the same manner as in an FSA actions associated with states are executed.

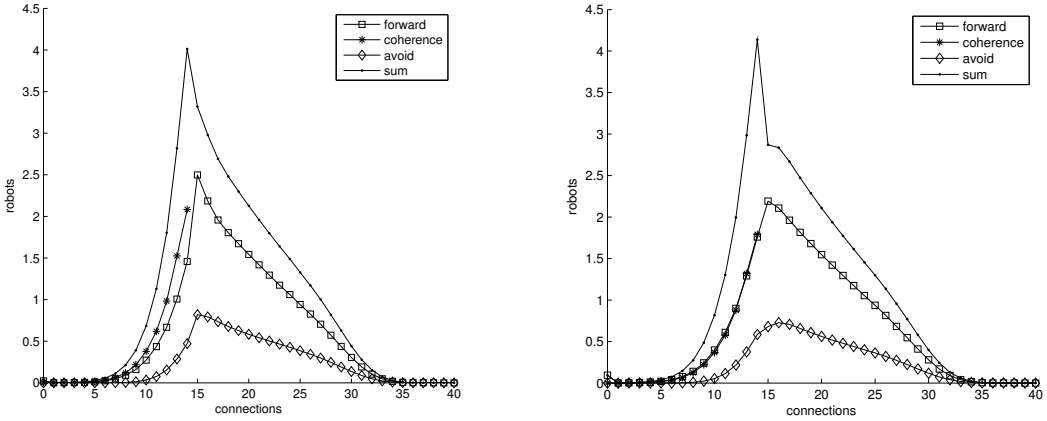
The original three state FSA is clearly of lower complexity than the final IAC for this case study. Nevertheless, this case study does not fully capture the intended level of complexity we envision for IR applications. We choose a simple case study in order to derive sound and clear conclusions for our conjectures before moving to more complex scenarios.

One of our goals is to assess the ability of our IAC approach to replicate results obtained with the original FSA approach. In order to do this, we performed realistic robotics simulations with both controllers described above. For performing simulations we used *Webots* [13], a sub-microscopic (every sensor and actuator represented separately with noise considered), flexible, 3D realistic simulator, and considered kinematic models of the *e-puck* robot [14], a differential drive robot of 7 cm in diameter with wireless communication capabilities. In this work, communication between *e-pucks* is not simulated realistically. We consider a perfect circular bounded communication radius and perfect package reception inside that radius (radial disk model).

We replicated the conditions of the original case study presented in [22], considering 40 robots in an unbounded arena performing the task over 10 000 seconds. The connection threshold was fixed to one single value, $\alpha = 15$, and the communication radius of the *e-puck* set to 0.7 m, instead of the original 2.0 m, in order to keep the ratio between communication and physical radius presented in the original paper.

To compare the performance of the original FSA controller and our IAC approach were performed 100 runs of the simulation for each implementation. Following a stability study for the task presented in [22], we chose to gather data only after 2 000 seconds of simulation on each run and average them over the remaining 8 000 seconds. This allowed us to avoid storing data from an initial transient period of the task. During simulations we stored the number of time steps spent in each state of the original controller ($forward$, $coherence$, and $avoid$) for each number of connections (between 0 and 40).

For the IAC simulations we must take into account that these states do not correspond to a single marked or non-marked place on the EPN but rather to one or more markings of the full net. For instance, state



(a) (b)

Figure 3: Average number of robots in each state plotted against connectivity in simulation: (a) original FSA controller; (b) IAC.

avoid corresponds to markings where the individual behavior $IndAv$ is active (macro place m_{IndAv} marked) and the *avoid* place is also marked. State *coherence* corresponds to all markings where institution $T180$ is active (macro place m_{T180} marked). All other markings correspond to state $forward$.

In Fig. 3 we display the average number of robots in each state plotted against connectivity (number of connections). The results are averaged over the 40 robots and all runs. Fig. 3-(a) shows results for the FSA controller simulations while Fig. 3-(b) shows results for the IAC approach. Results for state *coherence* are only available for connectivities below α (no robots in *coherence* have more than α connections). The black line corresponds to the sum of robots in all states. Results show a good agreement between the two approaches, both in number of robots and state distribution for each connectivity. The small differences between results are due to a small difference in the way both controllers update the neighborhood information. Results show a good agreement with those presented in the original paper [22].

5 Modeling and Performance Analysis Using Institutional Agent Controllers

In our research we are interested in borrowing some key concepts from a multi-level probabilistic modeling methodology established for swarm robotic systems [12], that takes into account the individual robot controller in order to generate an aggregated macroscopic representation of the dynamics of the whole team.

We use a Generalized Stochastic Petri Nets (GSPN) view of the original IAC [1] to construct our macroscopic model. In GSPN two types of transition are considered: immediate and (stochastically) timed. Timed

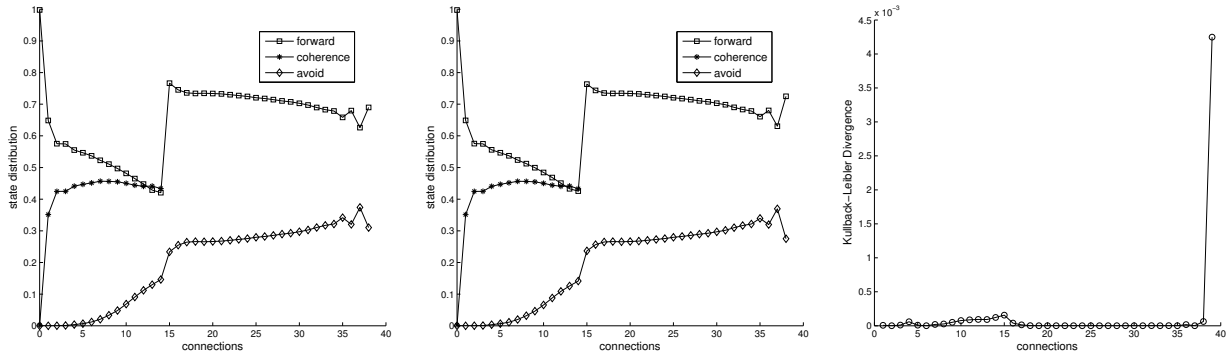
transitions specify that a probabilistically determined time interval must elapse before the transition can fire. If this time interval has a probabilistic exponential distribution with rate λ, we say that the transition is exponential with λ being called the transition rate. The marking process of a GSPN where all timed transitions are exponential is equivalent to a continuous time Markov chain [1]. This allows us to perform steady state and mean-field dynamics analysis of the GSPN (as one would with Markov chains). The states of our stochastic model represent the state of an "average robot", i.e., our model follows a mean-field approach, being representative of the whole team and not of a particular robot.

In this work, we are not concerned in modeling all details of our case study. Our goal is to establish that IACs can be used in modeling by providing the necessary structure to construct the desired GSPN models. Our approach is to use directly the EPN structure of the IAC to build our GSPN model. In order to do so, we need to identify what transitions correspond to timed transitions and estimate their rates. We focus on steady state analysis to assess the quality of the model. To do so, we use the software PIPE2 (Platform Independent Petri Nets Editor [2]) both for editing and performing steady state analysis of our models, based on closed form solution of Markov chain dynamics.

In [22], the multi-level probabilistic modeling methodology was applied to the wireless connected swarm case study. The FSA controller is used as a starting point for a Probabilistic FSA (PFSA) model that describes the swarm connectivity and overall state distribution of the swarm. In this work, we will focus on modeling the overall state distribution over the possible states, i.e., the average probability of a robot being in each state ($forward$, $coherence$, and $avoid$).

When applying the proposed modeling methodology to the institutional robotics approach we are mainly interested in studying the relationships between different behaviors specified in the higher layer of the IAC. However, in some cases, details about implementation of behaviors might also be of interest. By using the two layers of the IAC we can select sections of the EPN that are of interest to model a given system. For our proposed model, the higher layer would suffice if only states $forward$ and $coherence$ were considered (with $coherence$ corresponding to markings where the macro place for institution $T180$ is marked). However, to also consider state $avoid$ we need the lower layer implementation of the individual behavior, in order to make the distinction between $avoid$ and $forward$. The GSPN structure for our model is presented in Fig. 2-(b) as the section of the IAC encapsulated by the dashed line box.

The only immediate transitions in the model are those that are not associated with any condition. These correspond to the control transitions added during composition of behaviors, and are the transitions linking the macro place of the individual behavior m_{IndAv} with idle places $idle_{IndAv,T180}$ and $idle_{IndAv,TR}$. The remaining transitions in the model are stochastically timed and their transition rates need to be estimated. These correspond to conditions $obstacle$, $end\ avoid$, and the initial and final conditions for both institutions.



(a) (b) (c)

Figure 4: (a) State distribution predicted from GSPN model; (b) State distribution from simulation results; (c) Kullback-Leibler divergence between model and simulation results.

Given that our goal is only to establish that the IAC structure can be used as a GSPN model, we choose to estimate the transition rates directly from data gathered during the previously mentioned realistic simulations. The transition rates $\lambda_{i,k}$ are calculated separately for each timed transition i and each number of connections ($k = 0, \ldots, 40$) following:

$$\overline{t_{i,k}} = \frac{t_{i,k}}{N_R * N_{Runs} * T_{TS}}, \quad \overline{f_{i,k}} = \frac{f_{i,k}}{N_R * N_{Runs} * T_{TS}}, \quad \lambda_{i,k} = \frac{\overline{f_{i,k}}}{\overline{t_{i,k}}} \tag{3}$$

where $t_{i,k}$ is the number of total number of time steps the input places of transition i are marked and $f_{i,k}$ is the total number of times transition i fires, counted for all robots in all runs. T_{TS} is the number of time steps during one run, number of robots N_R is 40 and number of runs N_{Runs} is 100.

For each number of connections, we perform steady state analysis on our GSPN model with rates $\lambda_{i,k}$ for transitions *obstacle, end avoid*, $initial_{T180}$, $final_{T180}$, $initial_{TR}$ and $final_{TR}$. This analysis gives us the steady state probability for each tangible marking in our GSPN model. As before, we consider that state *avoid* corresponds to markings where the individual behavior is active and the *avoid* place is marked, state *coherence* corresponds to markings where institution $T180$ is active, and all other markings correspond to state $forward$. By summing the probabilities for all markings corresponding to each state we obtain the desired state distribution model. Note that for each number of connections k this model actually gives us a probability mass function for the discrete random variable describing possible states of the robots.

The model results are displayed in Fig. 4-(a). They show how the probability of being in each state varies with the number of connections k. In Fig. 4-(b), we display the state distribution from our simulations results (Fig. 3-(b)), by dividing the number of robots in each state for each k by the total number of robots for each connection gathered using simulations. In Fig. 4-(c) we display the Kullback-Leibler (KL) divergence between the probability mass functions given by the model for each k and the implicit probability distributions (for

each k) given by the data gathered from simulation. KL divergence between distributions p and q effectively measures the average likelihood of observing data with the distribution p if the particular model q actually generated the data [6]. We can observe an almost perfect matching between the two results. This is to be expected since our estimation of transition rates comes directly from the data gathered during the simulations. The larger error in Fig. 4-(c) comes from the extremely low number of time steps and transition fires for robots with 38 connections. This affects the calculation of the correct rate and thus generates a bigger error in the model. We conclude that with a correct estimation of transition rates, our IAC provides a good structure for the generation of macroscopic models.

6 Conclusion and Future Work

In this work we introduced an extension to the Petri Net formalism, Executable Petri Nets. These EPN have associated actions and conditions that allow them to be executed in robots through an algorithm presented in the paper. We defined institutions and an individual behavior for robots in a distributed robotic system making use of this new extension. In our approach, institutions are modular behaviors that can be modeled through an EPN and executed in a robot. Using a composition scheme controlled by dedicated deontic operators of a set of institutions we are able to obtain an Institutional Agent Controller (IAC) in the form on an EPN that combines several institutions and an individual behavior.

We also focused on assessing our proposed IAC approach. We implemented two versions of the wireless connected swarm case study. The original version of the case study uses an FSA controller. The second version uses an IAC composed of one individual behavior and two institutions designed to perform the task similarly. Realistic simulation results show a good agreement between the two versions. We can conclude that our IAC approach is able to replicate results produced with other control approaches.

We also assessed the quality of modeling distributed robotic systems by EPNs. Using the EPN structure of the IAC designed for the case study, we were able to construct a GSPN model for the overall state distribution of the system. By using data gathered from realistic simulations in order to estimate the transition rates necessary for our GSPN model, we were able to observe a very good agreement between model predictions and simulation results. We can conclude that with a correct estimation of transition rates, our IAC provides a good structure for the generation of macroscopic models.

In relatively simple case studies, as the one discussed, the benefits of the IR approach may not be easily noticeable. However, under more complex tasks and scenarios of strong social nature, possibly populated by humans, the IR approach will provide "natural" coordination artifacts to achieve flexible coordination among all actors. Considering, for instance, the task of autonomous navigation on a road shared with human

drivers, formalizing traffic rules as social institutions will simplify the coordination burden of the task, since the autonomous vehicle will not have to deliberate on what to do in the presence of traffic signs and driving options, and will provide a natural way to interact with other actors.

In the future we plan to apply the IAC methodology to such social complex scenarios so as to demonstrate their ability to coordinate and analyze robot societies following IR principles. We also intend to further improve our GSPN models, for instance, with estimation of transition rates directly computed using geometrical considerations of the scenario rather than on data gathered in simulation. An alternative algorithm for this case study, considering the sharing of neighborhood information among robots, is presented in [15]. We intend to design an IAC for this algorithm and apply our modeling methodology, in order to investigate situations where the microscopic-to-macroscopic (or individual-to-swarm) mapping might be less straightforward to capture accurately because of the additional complexity of the coordination algorithm.

7 Acknowledgments

This work was partially supported by Fundação para a Ciência e a Tecnologia (FCT) through grants SFRH/BD/33671/2009 (first author) as part of the Joint Doctoral Program IST-EPFL and SFRH/BPD/35862/2007 (second author), as well as by FCT ISR/IST Pluriannual funding through the PIDDAC program funds.

References

[1] Bause, F. and P. S. Kritzinger (2002, August). *Stochastic Petri Nets - An Introduction to the Theory* (Second ed.). Braunschweig/Wiesbaden (Germany): Friedr. Vieweg &Sohn Verlag.

[2] Bonet, P., C. Lladó, R. Puijaner, and W. Knottenbelt (2007). PIPE v2.5: A Petri net tool for performance modelling. In *Proc. 23rd Latin American Conference on Informatics (CLEI 2007)*, San Jose, Costa Rica.

[3] Cassandras, C. G. and S. Lafortune (2008). *Introduction to Discrete Event Systems* (Second ed.). Springer.

[4] Correll, N. and A. Martinoli (2011). Modeling and designing self-organized aggregation in a swarm of miniature robots. *International Journal of Robotics Research 30*(5), 615–626.

[5] Costelha, H. and P. Lima (2012, March). Robot task plan representation by Petri nets: modelling, identification, analysis and execution. *Autonomous Robots*, DOI: 10.1007/s10514–012–9288–x.

[6] Cover, T. M. and J. A. Thomas (2006). *Elements of Information Theory, Second Edition.* New York: Wiley.

[7] Crawford, S. E. S. and E. Ostrom (1995). A Grammar of Institutions. *The American Political Science Review 89*(3), 582–600.

[8] Dias, M., R. Zlot, N. Kalra, and A. Stentz (2006, July). Market-Based Multirobot Coordination: A Survey and Analysis. *Proceedings of the IEEE (Special Issue on Multirobot Coordination) 94*(7), 1257–1270.

[9] Hahn, C., B. Fley, M. Florian, D. Spresny, and K. Fischer (2007). Social reputation: A mechanism for flexible self-regulation of multiagent systems. *Journal of Artificial Societies and Social Simulation 10*(1).

[10] Hexmoor, H., S. G. Venkata, and D. Hayes (2006, March). Modelling social norms in multiagent systems. *Journal of Experimental & Theoretical Artificial Intelligence 18*(1), 49–71.

[11] Hodgson, G. M. (2000). What Is the Essence of Institutional Economics ? *Journal of Economic Issues 34*(2), 317–329.

[12] Martinoli, A., K. Easton, and W. Agassounon (2004). Modeling swarm robotic systems: A case study in collaborative distributed manipulation. *Int. Journal of Robotics Research 23*(4), 415–436.

[13] Michel, O. (2004). Webots TM : Professional Mobile Robot Simulation. *Advanced Robotic 1*(1), 39–42.

[14] Mondada, F., M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J. Zufferey, D. Floreano, and A. Martinoli (2009). The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions*, Volume 1, pp. 59–65.

[15] Nembrini, J., A. F. T. Winfield, and C. Melhuish (2002). Minimalist Coherent Swarming of Wireless Networked Autonomous Mobile Robots. *From animals to animats 7: Proceedings of the seventh international conference on simulation of adaptive behavior*, 273–282.

[16] Omicini, A., A. Ricci, M. Viroli, C. Castelfranchi, and L. Tummolini (2004). Coordination artifacts: Environment-based coordination for intelligent agents. In *3rd international Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004)*, New York, NY, USA, pp. 286–293.

[17] Ricci, A., M. Viroli, A. Mater, and S. Università (2005). Coordination Artifacts : A Unifying Abstraction for Engineering Environment-Mediated Coordination in MAS. *Informatica 29*, 433–443.

[18] Sabater, J. and C. Sierra (2005, September). Review on Computational Trust and Reputation Models. *Artificial Intelligence Review 24*(1), 33–60.

[19] Silva, P. and P. U. Lima (2007). Institutional Robotics. In *Proc. of ECAL 2007 - 9th European Conference on Artificial Life*, Lisboa, Portugal, pp. 157–164.

[20] Silva, P., R. Ventura, and P. U. Lima (2008, December). Institutional environments. In *Proc. of Workshop AT2AI: From agent theory to agent implementation, AAMAS 2008 - 7th International Conference on Autonomous Agents and Multiagent Systems*, Volume 35, Estoril, Portugal, pp. 595–604.

[21] Tummolini, L. and C. Castelfranchi (2006, June). The cognitive and behavioral mediation of institutions: Towards an account of institutional actions. *Cognitive Systems Research 7*(2-3), 307–323.

[22] Winfield, A. F. T., W. Liu, J. Nembrini, and A. Martinoli (2008, September). Modelling a wireless connected swarm of mobile robots. *Swarm Intelligence 2*(2-4), 241–266.

[23] Ziparo, V. A., L. Iocchi, P. U. Lima, D. Nardi, and P. F. Palamara (2011). Petri Net Plans: A Framework for Collaboration and Coordination in Multi-Robot Systems. *Autonomous Agents and Multi-Agent Systems 23*(3), 344–383.

Algorithm 2 Execute Petri Net

```
repeat
    for all enabled transitions t_i ∈ T do
        if associated condition c_i is true then
            run associated parameter action pa_i
            fire transition t_i
        end if
    end for
until no transition has fired
for all marked places p_i ∈ P do
    run associated action a_i
end for
```

Figure: *Execute Petri Net* Algorithm.

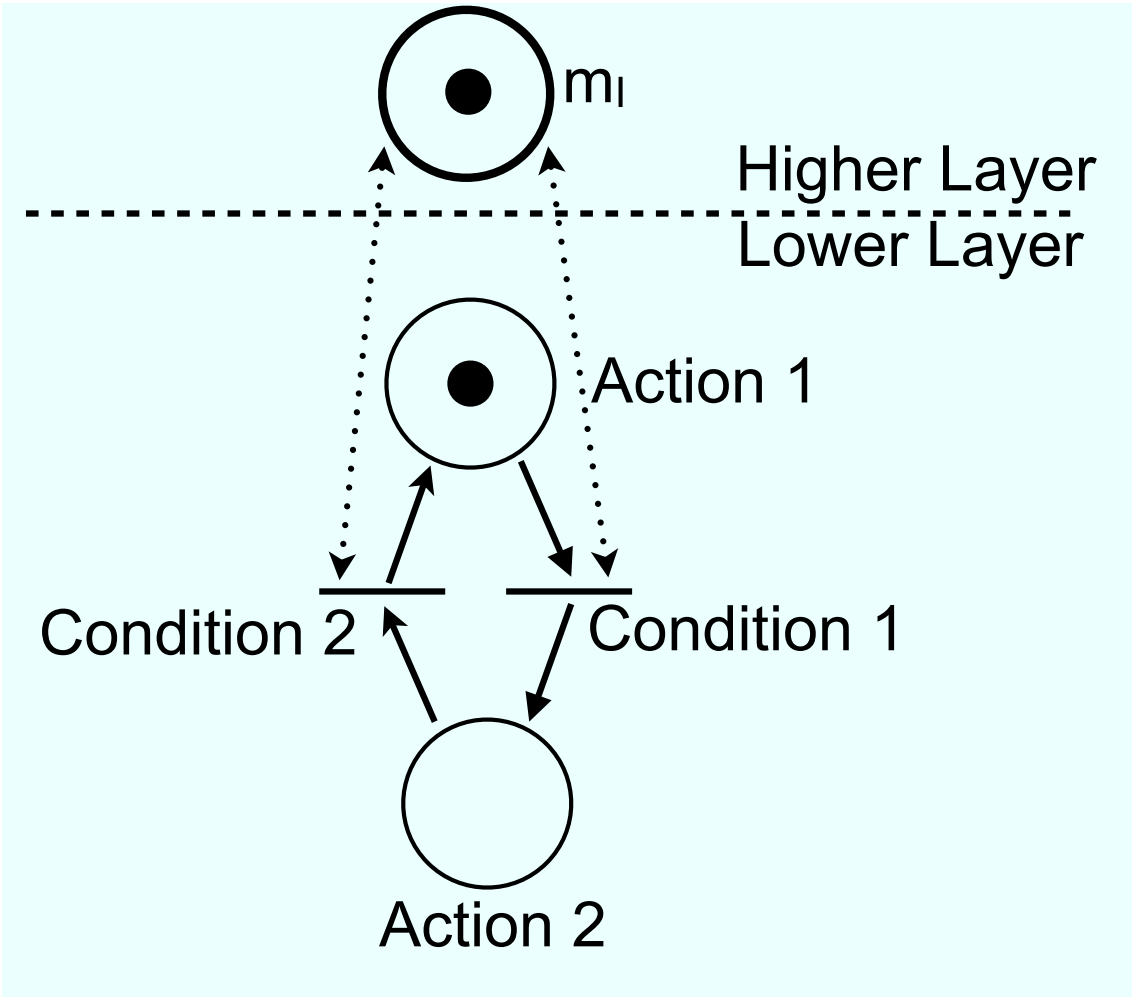


Figure 5: Hierarchical representation of an EPN in two layers. Dotted arcs represent two directional arcs, one from a transition to a place and one from a place to a transition. Bottom: lower layer, EPN $Inst$ with conditions and actions associated to transitions and places. Top: higher layer, macro place m_I.

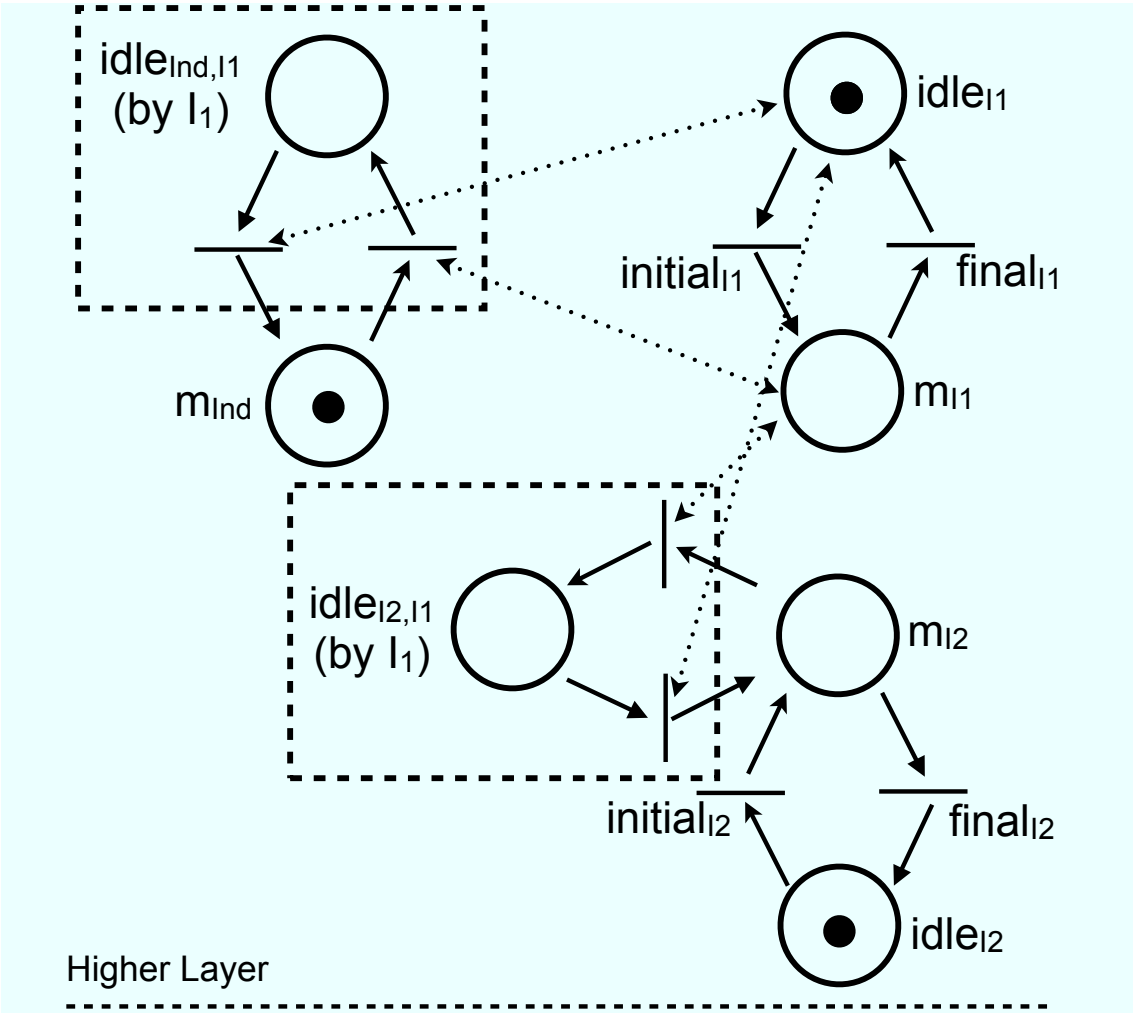


Figure 6: Higher layer composition scheme for two institutions I_1, I_2 and individual behavior Ind. Dotted arcs represent bidirectional arcs, as in Fig. 1. Places m_{I1}, m_{I2}, m_{Ind} are higher layer macro places representing implementations of institutions and the individual behavior in the lower layer. These representations will be used throughout the paper. Places and transitions encapsulated by dashed line boxes are added during composition procedure.

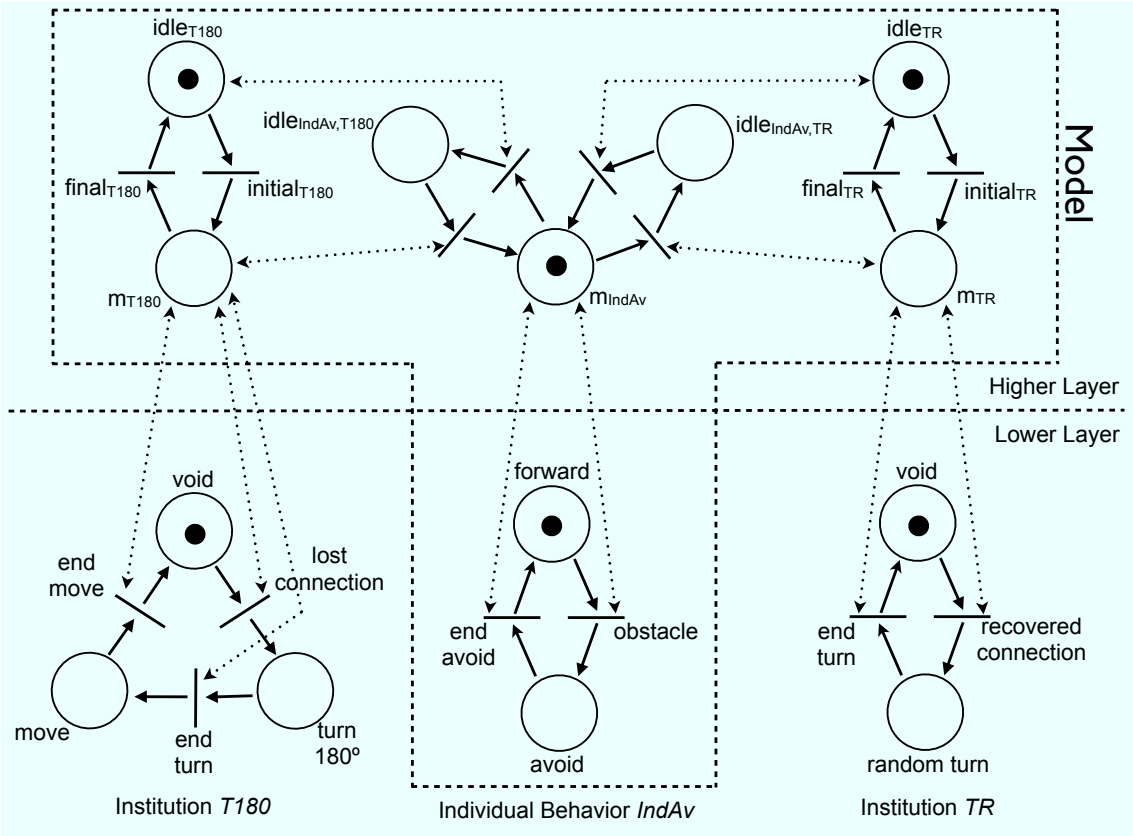


Figure 7: IAC for wireless connected swarm. Lower layer: EPNs for individual behavior $IndAv$ and institutions $T180$ and TR. Higher layer: composition of individual behavior and institutions. PN structure for GSPN model encapsulated in dashed line box.

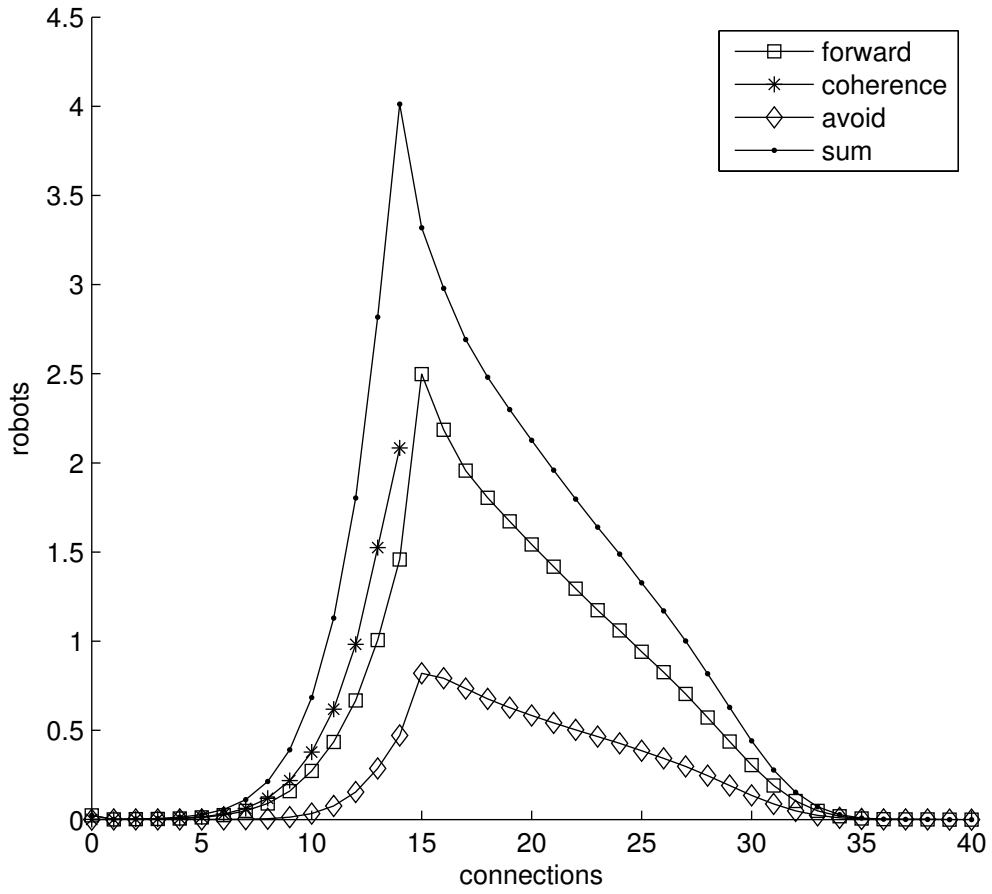


Figure 8: Average number of robots in each state plotted against connectivity in simulation: original FSA controller.

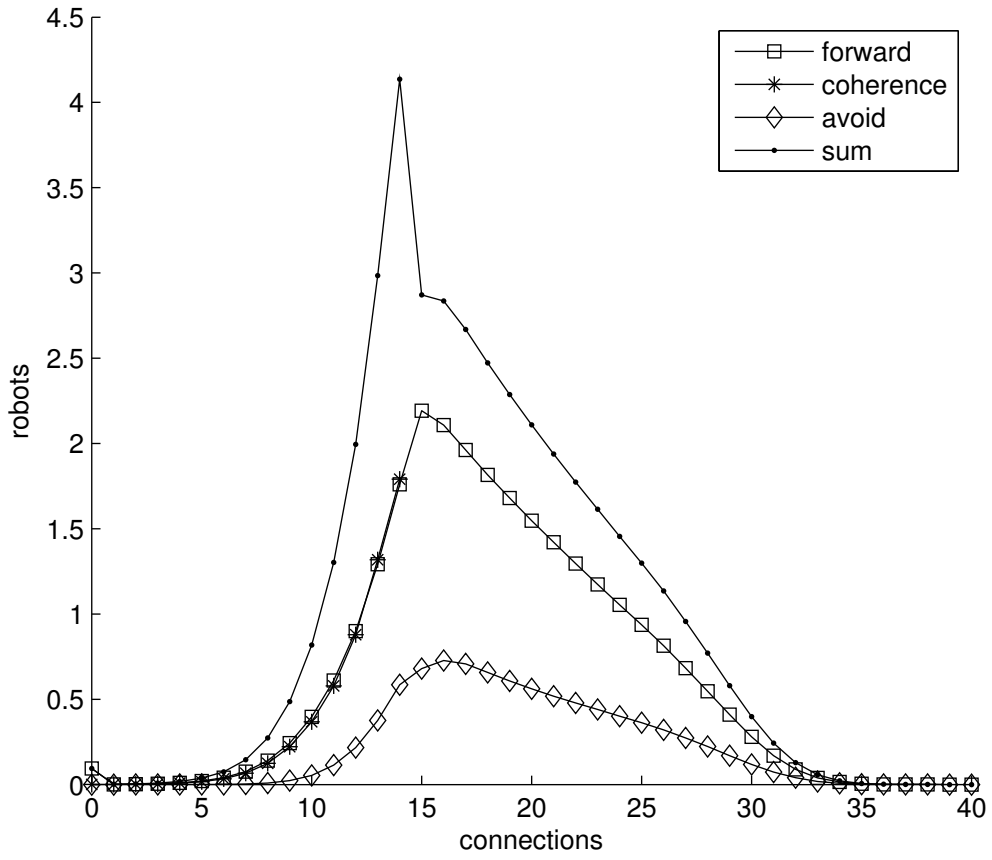


Figure 9: Average number of robots in each state plotted against connectivity in simulation: IAC.

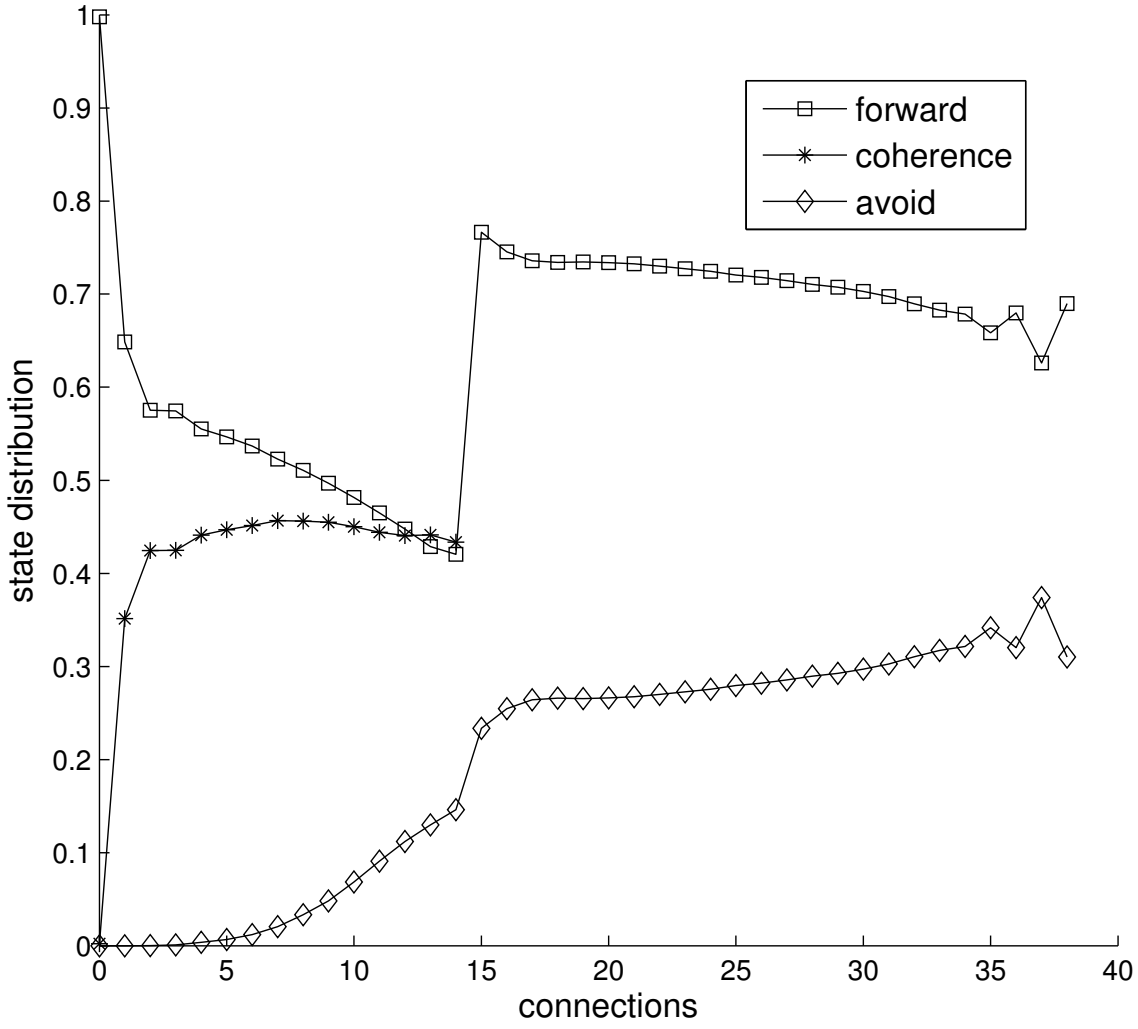


Figure 10: State distribution predicted from GSPN model.

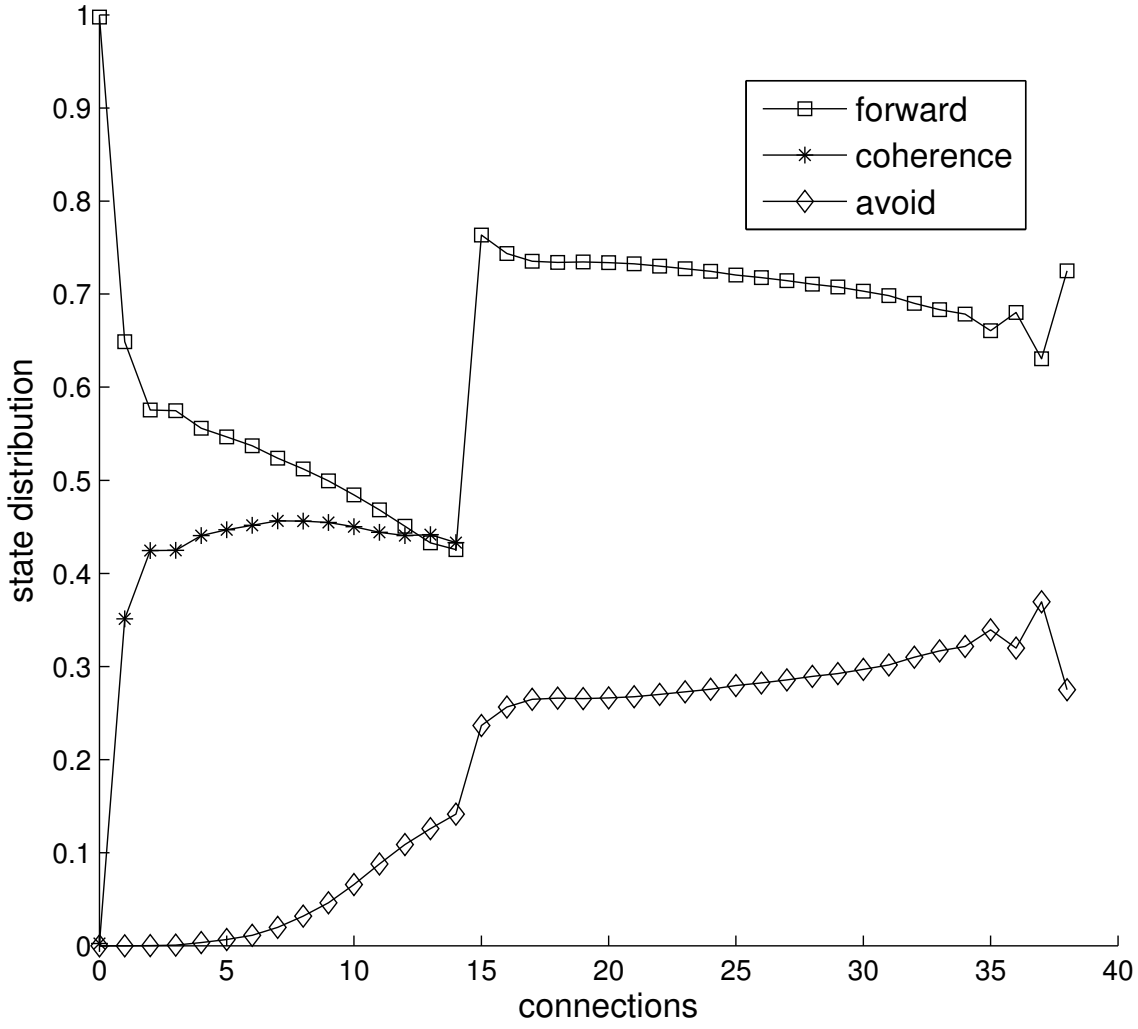


Figure 11: State distribution from simulation results.

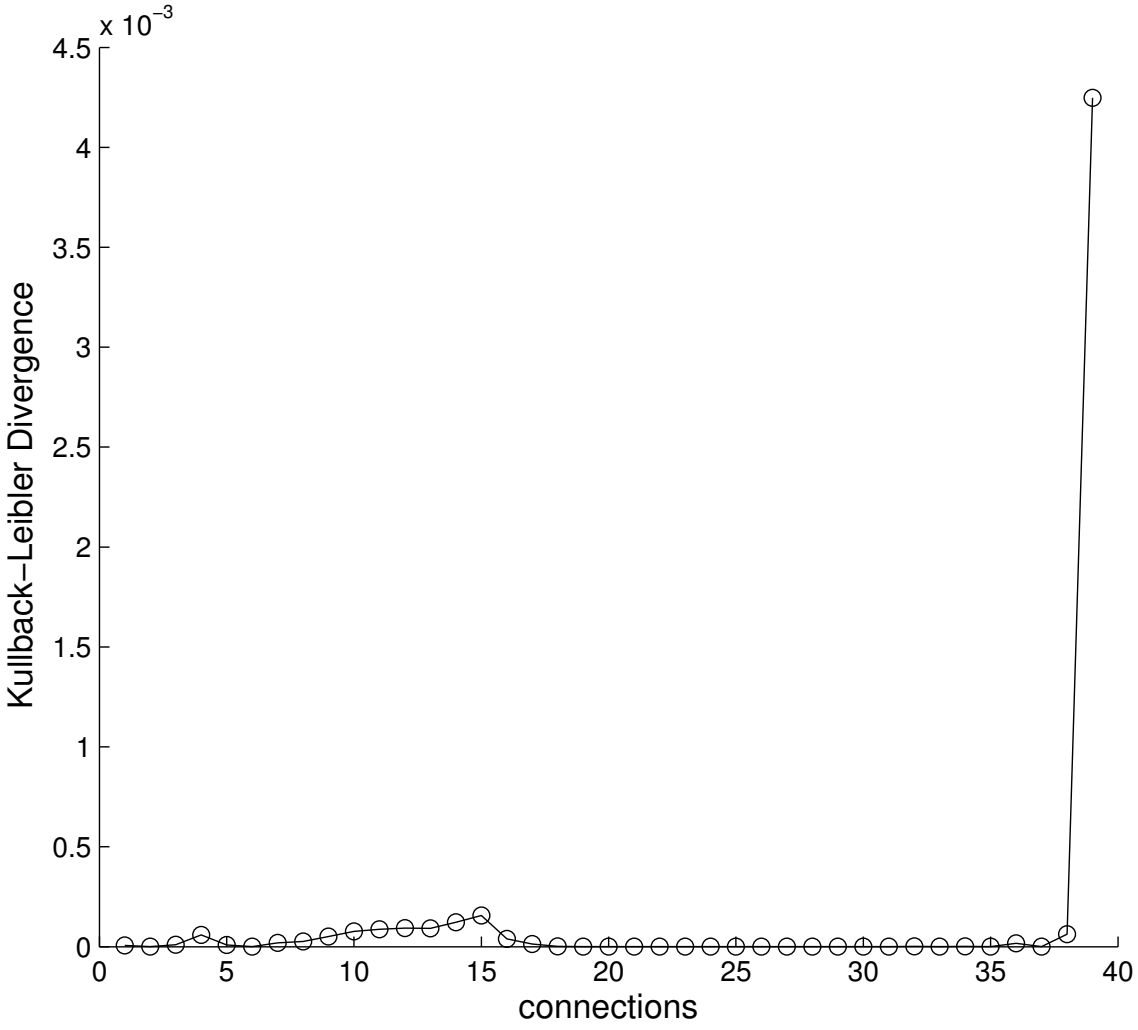


Figure 12: Kullback-Leibler divergence between model and simulation results.