Chapter 13
Markovian Agent Models: A Dynamic Population of Interdependent Markovian Agents

Andrea Bobbio, Davide Cerotti, Marco Gribaudo, Mauro Iacono and Daniele Manini

Abstract A Markovian Agent Model (MAM) is an agent-based spatio-temporal analytical formalism aimed to model a collection of interacting entities, called Markovian Agents (MA), guided by stochastic behaviours. An MA is characterized by a finite number of states over which a transition kernel is defined. Transitions can either be local, or induced by the state of other agents in the system. Agents operate in a space that can be either continuous, or composed by a discrete number of locations. MAs may belong to different classes and each class can be parametrized depending on the location in the geographical (or abstract) space. In this work, we provide a very general analytical formulation of an MAM that encompasses many forms of physical dependencies among objects and many ways in which the spatial density may change in time. We revisit recent literature to show how previous works can be cast in terms of this more general MAM formulation.

Keywords Agent-based model · Spatially distributed systems · Performance modelling

13.1 Introduction

A Markovian Agent Model (MAM) is an agent-based spatio-temporal analytical formalism aimed to model a collection of interacting Markovian Agents (MAs). An MA has a finite number of possible discrete operating modes (states) over which

A. Bobbio
Università del Piemonte Orientale, Alessandria, Italy

D. Cerotti · M. Gribaudo (✉)
Politecnico di Milano, Milan, Italy
e-mail: marco.gribaudo@polimi.it

M. Iacono
Seconda Università di Napoli, Caserta, Italy

D. Manini
Università di Torino, Turin, Italy

a transition kernel is defined. The transition kernel is composed by two components a *local transition matrix* and an *induced transition matrix*. The local matrix contains a fixed component that depends on the MA structure and its position in the space but does not depend on the interaction with the other MAs. The induced matrix depends on the interaction of an MA with the other MAs. MAs may belong to different classes and each class has a spatial density function that depends on the location in the geographical (or abstract) space. In this work, we provide the solution equations for a very general MAM that encompasses a *static* MAM in which the spatial density of the MAs of any class remains constant in time and a *dynamic* MAM in which the spacial density varies as a function of the time. The density variation in the dynamic MAM can be due to the displacement of the MAs in the space (being constant their overall number) or to the birth or death of MAs. A typical example of a population of agents with similar characteristics is represented by the number of calls in a wireless cellular network where each cell can have a different number of ongoing calls, and this number may change due to new calls (birth), completion of calls (death) or and handoff-in and handoff-out calls from adjacent cells, or also due to reduction in the performance of a cell caused by degradation or failure [30].

The modelling and analysis of large-scale stochastic systems composed by spatially defined interacting objects has been mainly faced in the literature by resorting to the superposition of interacting Markov chains [1, 25], to stochastic Reward model or hierarchical models [10, 23], to various process algebra formalisms [15, 20] and to fluid models [18, 21].

In recent years, the MAM has emerged as a new versatile analytical technique whose main idea is to model a distributed system by means of interacting agents, so that each agent maintains its local properties but at the same time modifies its behaviour according to the influence of the interaction with the other agents [3, 17, 20]. By separating the local behaviour of an MA, that does not depend on the interaction with other agents, from the influence of the other MAs, we can avoid the construction of the combined state space. The solution of the overall model is obtained by building several models, one for each MA, and then solving them separately. This technique, that avoids generating a large model, is sometimes referred to as *largeness avoidance*. In the present work, we provide a very general analytical formulation of an MAM that encompasses many kinds of forms of physical dependencies among objects and many ways in which the spatial density may change in time. We revisit recent literature to show how previous works can be cast in terms of this more general MAM formulation.

Section 13.2 introduces the theory of MAM to arrive to the equations describing the dynamics of the probability distribution of a class of MAs as a function of the time and of the MA location in the space. Section 13.3 discusses the way in which the influence matrix is defined and evaluated for several different case studies. A wide spectrum of different cases is discussed in this section to give the reader a flavour of the possibilities offered by the technique.

13.2 Theory

Markovian Agents describe systems composed by several interacting agents, each one characterized by a set of possible *states*, and whose evolution can be described by the way in which they evolve in their state space. MAs behaviour is similar to Automata, and in particular to Continuous Time Markov Chains (CTMCs). The structure of a single MA is represented in Fig. 13.1. The circles labelled $i, j, \ldots k$, represent the states of the CTMC describing the MA. Differently from conventional CTMC models, the transitions among the states are of two possible types that are drawn differently:

- Solid lines (like the transition from i to j or the self-loops in i or in j) represent the local or autonomous behaviour of the objects. They are the fixed component of the infinitesimal generator that are independent of the interaction with the other MAs. For instance, they can be used to model a failure occurring to the agent, or the reaction to an internal stimulus. MAs include also self-loop transitions that require a particular notation since they are not visible in the infinitesimal generator of the CTMC [29]. Self-loop transitions are required because, even if they do not vary the state of the agent, they might influence the behaviour of other agents.
- Dashed lines (like the transition from i to k or the transitions into i or j) represent the transitions induced by the interaction with the other MAs. In the literature, several different types of induction have been considered. In any case, induction, as the name suggests, *induces* an agent to change state at a rate that depends on the state of the other agents. The way in which the rates of the induced transitions are computed is explained in Sect. 13.3.

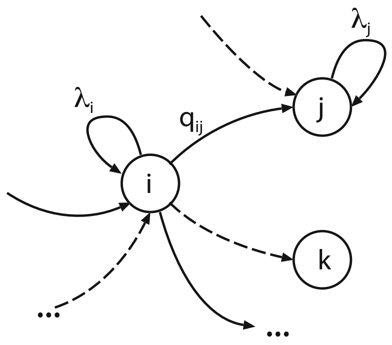

Fig. 13.1 Schematic structure of a Markovian agent

13.2.1 Markovian Agents Model

The Markovian Agent Model (MAM) represents a system as a collection of *Markovian Agents* (MAs) spread over a geographical space $\mathcal{V}$. As introduced, the essence of the MAM is that each MA is described by a CTMC, whose infinitesimal generator contains a fixed component that depends on the MA structure and position in space $\mathbf{v} \in \mathcal{V}$, and a component that depends on the interaction with the other MAMs.

Space $\mathcal{V}$ can be either discrete or continuous. If discrete, $\mathcal{V} = \{\mathbf{v_1}, \mathbf{v_2}, \ldots \mathbf{v_N}\}$ where $\mathbf{v_i}$ are the locations. If continuous, $\mathcal{V} \subseteq \mathbb{R}^k$, with $1 \leq k \leq 3$. Agents are divided in C classes, and an MA belonging to class c (with $1 \leq c \leq C$) is characterized by n_c different states. Let us call $p_j^{\{c\}}(t, \mathbf{v})$ the probability that a class c agent is in state $1 \leq j \leq n_c$ at time t, at location $\mathbf{v} \in \mathcal{V}$.

Agents are analyzed using *counting process* and exploiting mean field approximation [2, 24]. In particular, any location $\mathbf{v}$ can hold a number of class c agents that is defined by function $\rho_c(t, \mathbf{v})$. For discrete space models, $\rho_c(t, \mathbf{v})$ accounts for the average number of class c agents in location $\mathbf{v}$ at time t. For continuous space models, $\rho_c(t, \mathbf{v})$ is the average density of class c agents in point $\mathbf{v}$ at time t, and for any finite area $\Omega \subseteq \mathcal{V}$, the average number $N_c(t, \Omega)$ of class c agents contained inside Ω at time t can be computed as

$$N_c(t, \Omega) = \int_{\Omega} \rho_c(t, \mathbf{v}) d\mathbf{v}.$$

We also define $\boldsymbol{\pi}_j^{\{c\}}(t, \mathbf{v}) = p_j^{\{c\}}(t, \mathbf{v}) \cdot \rho_c(t, \mathbf{v})$ as the density of class c agents in state j at location $\mathbf{v}$ and time t. Note that if we are considering a discrete model, where each location has exactly one agent, we have $\pi_j^{\{c\}}(t, \mathbf{v}) = p_j^{\{c\}}(t, \mathbf{v})$. Value $\pi_j^{\{c\}}(t, \mathbf{v})$ will be the main performance index that will be exploited to compute all the interesting measures in the model solution process.

According to the definition of the density $\rho_c(t, \mathbf{v})$, we can then classify MAMs with the following taxonomy:

- An MAM is *static* if $\rho(t, \mathbf{v})$ does not depend on time, and *dynamic* otherwise;
- An MAM is *discrete* if the geographical area on which the MAs are deployed is discretized and $\rho(t, \mathbf{v})$ is a discrete function of the space or it is *continuous* if $\rho(t, \mathbf{v})$ is a continuous function of the space;
- An MAM is *single class* if $C = 1$, and *multi class* if $C > 1$.

To simplify the notation, let us collect the terms into row vectors $\pi_c(t, \mathbf{v}) = |\pi_j^{\{c\}}(t, \mathbf{v})|$ that represents the state distribution of an MA belonging to class c at time t in position $\mathbf{v}$. Moreover, let $\Pi_{\mathcal{V}}(t) = \{(c, \mathbf{v}, \pi_c(t, \mathbf{v})) : 1 \leq c \leq C, \mathbf{v} \in \mathcal{V}\}$ be the ensemble of the probability distribution of all the agents of all the classes at time

t, and let us denote with $[\Pi_{\mathcal{V}}] = \{(t', \Pi_{\mathcal{V}}(t')) : 0 \leq t' < t\}$ the evolution of the ensemble of the probability distribution from the initial state up to time t. The dynamic of the state density distribution of a class c agent in position $\mathbf{v}$ is described by the following equation:

$$\frac{\partial \pi_c(t, \mathbf{v})}{\partial t} + \frac{\partial(\boldsymbol{\omega}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) \cdot \pi_c(t, \mathbf{v})^T)}{\partial \mathbf{v}} = \nu_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) + \pi_c(t, \mathbf{v}) \cdot \mathbf{K}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}]). \tag{13.1}$$

The terms $\boldsymbol{\omega}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}])$, $\boldsymbol{\nu}(t, \mathbf{v}, [\Pi_{\mathcal{V}}])$ and $\mathbf{K}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}])$ are respectively the *motion*, *increase* and *transition* kernels. They can all depend on the class c, on the position $\mathbf{v}$, on the time t and on the ensemble probability $[\Pi_{\mathcal{V}}]$, and they rule the evolution of the system.

The motion kernel $\boldsymbol{\omega}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}])$ is used only for agents deployed in continuous space: for models that are characterized by a finite number of locations, the second term on l.h.s. of Eq. (13.1) can be discarded since it is not required. Otherwise $\boldsymbol{\omega}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) = (\boldsymbol{\omega}_x^{\{c\}}(t, \mathbf{v}, [\Pi_{\mathcal{V}}]), \boldsymbol{\omega}_y^{\{c\}}(t, \mathbf{v}, [\Pi_{\mathcal{V}}]), \ldots)$ is a set of diagonal matrices whose element $\boldsymbol{\omega}_x^{\{c:i\}}(t, \mathbf{v}, [\Pi_{\mathcal{V}}])$ represents the speed along direction x for a class c agent in state i (with $1 \leq i \leq n_c$). With a slight abuse of notation, $\frac{\partial}{\partial v} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \ldots\right)$ denotes the partial derivates along all the dimensions of space V. For example, for a two-dimensional system, we have:

$$\begin{aligned}\frac{\partial(\boldsymbol{\omega}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) \cdot \pi_c(t, \mathbf{v})^T)}{\partial v} = & \frac{\partial(\boldsymbol{\omega}_x^{\{c\}}(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) \cdot \pi_c(t, \mathbf{v})^T)}{\partial x} \\ & + \frac{\partial(\boldsymbol{\omega}_y^{\{c\}}(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) \cdot \pi_c(t, \mathbf{v})^T)}{\partial y}.\end{aligned} \tag{13.2}$$

The increased kernel $\boldsymbol{\nu}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}])$ accounts for the effects that grows the number or the density of agents in a point in space. It can be subdivided into two terms:

$$\boldsymbol{\nu}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) = b_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) + m_c^{in}(t, \mathbf{v}, [\Pi_{\mathcal{V}}]). \tag{13.3}$$

Term $b_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}])$ is the *birth* term: it is used to model the autonomous generation of agents and expresses the rate (measured in agents per time unit or agent density per time unit) at which class c agents are created in location $\mathbf{v}$ at time t. Term $\mathbf{m}_c^{in}(t, \mathbf{v}, [\Pi_{\mathcal{V}}])$ is the location *input* term, and accounts for class c agents that "warp" into location $\mathbf{v}$ at time t from other points in space. In discrete space model, it is used to model cell transfer of agents. Continuous space models may use this term to account for "wormhole" or "portal" effects where agents that reach one location, can vanish and immediately reappear in other points in the space.

The transition kernel $\mathbf{K}_c(t, \mathbf{v}, [\Pi_\mathcal{V}])$ accounts for both the state transitions of the agents, and for the effects that can reduce the number of agents in one location. It can be subdivided into four terms:

$$\begin{aligned}\mathbf{K}_c(t, \mathbf{v}, [\Pi_\mathcal{V}]) = \mathbf{Q}_c(t, \mathbf{v}) + I_c(t, \mathbf{v}, [\Pi_\mathcal{V}]) + \\ - \mathbf{D}_c(t, \mathbf{v}, [\Pi_\mathcal{V}]) - \mathbf{M}_c^{out}(t, \mathbf{v}, [\Pi_\mathcal{V}]).\end{aligned} \tag{13.4}$$

Matrix $\mathbf{Q}_c(t, \mathbf{v}) = |q_{ij}^{\{c\}}(t, \mathbf{v})|$ in Eq. (13.1) defines the rate of *local transitions*, as in traditional CTMC, according to the agent position $\mathbf{v}$ and time t. Note that to further emphasize the locality of this term, $\mathbf{Q}_c(t, \mathbf{v})$ does not depend on the complete state history of the model $[\Pi_\mathcal{V}]$. The influence matrix $\mathcal{I}_c(t, \mathbf{v}, [\Pi_\mathcal{V}])$ accounts for the rate of *induced transitions* due to the influence of other agents. The entries of matrix $\mathcal{I}_c(t, \mathbf{v}, [\Pi_\mathcal{V}])$ depend on the state probabilities of other agents and must satisfy precise structural restrictions so that the matrix $\mathbf{Q}_c(t, \mathbf{v}) + \mathcal{I}_c(t, \mathbf{v}, [\Pi_\mathcal{V}])$ is still an infinitesimal generator matrix. Matrix $\mathbf{D}_c(t, \mathbf{v})$ is the *death* term: it is a diagonal matrix whose elements $d_{ii}^{\{c\}}(t, \mathbf{v})$ represent the rate at which class c agent in location $\mathbf{v}$ at time t may disappear from the model depending on its state i. Matrix $\mathbf{M}_c^{out}(t, \mathbf{v}, [\Pi_\mathcal{V}])$ is a matrix whose terms $m_{ij}^{out:\{c\}}(t, \mathbf{v})$ account for the discrete output from a location $\mathbf{v}$ at time t for a class c agent. **If** $i = j$, output does not cause a change of state; otherwise, the state of the agent may vary during its motion. It is the exit counterpart of vector $\mathbf{m}_c^{in}(t, \mathbf{v}, [\Pi_\mathcal{V}])$ previously introduced. In particular, the two terms are usually related by some conservation law. For example, if agents can warp from location $\mathbf{v}$ to location $\mathbf{u}$ at rate λ, we have $\mathbf{m}_c^{in}(t, \mathbf{u}, [\Pi_\mathcal{V}]) = |\lambda, \ldots|\pi_c(t, \mathbf{v})$ and $\mathbf{M}_c^{out}(t, \mathbf{v}, [\Pi_\mathcal{V}]) = diag(\lambda, \ldots)$.

In order to solve the model, the initial state of the system must be provided. For discrete space models, it is enough to provide $\rho_c(0, \mathbf{v})$, the initial density of class c agents in location $\mathbf{v}$, and $p_j^{\{c\}}(0, \mathbf{v})$, the corresponding initial state probability. From $p_j^{\{c\}}(0, \mathbf{v})$ and $\rho_c(0, \mathbf{v})$ we can then compute the initial condition of Eq. (13.1), and express it as:

$$\pi_j^{\{c\}}(\mathbf{v}, \nu) = p_j^{\{c\}}(0, \mathbf{v}) \cdot \rho_c(0, \mathbf{v}). \tag{13.5}$$

Continuous space models also require boundary conditions to describe what happens to agents that exit the domain V, and whether external agents enters inside V from outside. To simplify the presentation, we will limit our discussion to cases in which agents are confined inside V. In this case the boundary conditions are:

$$\rho_c(t, \mathbf{v}) = 0, \forall \mathbf{v} \in \mathbf{Boundary}(\mathcal{V}), \tag{13.6}$$

where $\mathbf{Boundary}(\mathcal{V})$ is the set of all points that define the boundary of the considered space $\mathcal{V}$.

13.3 Induced Transitions

Equation (13.1) provides an abstract description of the agents' behaviour, since the rates of induced transitions composing the influence matrix are not fully specified. The definition and evaluation of the influence matrix depend on the considered problem. In this section, we will present several cases from the literature, and we will show how they can be incorporated in Eq. (13.1).

13.3.1 Largeness Avoidance: Shared Repair in Complex Systems

Large fault-tolerant systems are composed by many subsystems that may undergo failures and repairs during operation. High availability is a prescribed requirement in many applications and necessitates an accurate model. To capture dynamic behaviour and dependencies that arise in the failure and repair process of the system, a preferred way is to resort to state-space models. However, state-space models suffer from state-space explosion; i.e. extremely large state-space is required for the accurate modelling of real systems, referred to as *largeness*. A way to tackle largeness is to avoid generating a large model, called *largeness avoidance*. Equation 13.1 offers a technique of *largeness avoidance*. Each subsystem is modelled as an MA with its local infinitesimal generator $\mathbf{Q}_c(t)$ (there is no spatial dependence on the position $\mathbf{v}$, in this case) and the influence matrix $\mathcal{I}_c(t, [\Pi_{\mathcal{V}}])$ that must be inferred from the physical dependencies inside the system. The *largeness avoidance* is obtained by computing the overall system dependability measures by solving individual MAs defined on the state-space of each subsystem. Shared repair is an usual source of dependence, and examples of availability modelling with various kinds of dependencies due to shared repair are given in [27, 28].

A simple example, inspired from [27], is the following. A system is composed by n subsystems that for simplicity we represent as a two-state model with a single up state u and a single down state d. The subsystems share a single repair person that follows a *preemptive repair priority policy*; i.e. the subsystems are ordered according to a predefined repair list, that for the sake of simplicity we assume in the natural order $X_1 \succ X_2 \succ \cdots \succ X_n$, and upon failure the repair crew starts repairing the subsystems with higher priority (subsystem X_i before X_j with $j > i$). Let us call λ_i the failure rate of component i, and μ_i the corresponding repair rate. We consider here only the steady-state solution.

Given that X_1 is repaired first, the repair of the other subsystems with lower priority is delayed. We can account for this delay modifying the repair rates of the subsystems with lower priority. The MA related to model X_1 is solved first and the probability $\pi_d^{X_1}$ that the subsystem X_1 is under repair (i.e. is in the down state) is calculated. Thus the probability that the repair person is idle is $\pi_u^{X_1} = (1 - \pi_d^{X_1})$. Subsystem X_2 can be repaired only if the repair person is not busy with subsystem

X_1 (i.e. X_1 is in the up state), which can be accounted for by reducing the repair rate of subsystem X_2 by the factor $\pi_u^{X_1}$:

$$\mu_2' = \mu_2 \pi_u^{X_1} = \mu_2 (1 - \pi_d^{X_1}).$$

By the same reasoning, subsystem X_j can be repaired only if all the subsystems $X_i, (i<j)$ are not under repair and are operating. This effect is accounted for by reducing the repair rate of subsystem X_j by:

$$\mu_j' = \mu_j \prod_{i<j} \pi_u^{X_i} = \mu_j \prod_{i<j} (1 - \pi_d^{X_i}).$$

The above problem can be rewritten in terms of MAM using Eq. (13.4). Each subsystem is an MA whose local kernel contains only the failure transitions that are independent of the rest of the system:

$$\mathbf{Q}^{X_j} = \begin{bmatrix} -\lambda_j & \lambda_j \\ 0 & 0 \end{bmatrix} \quad \text{for} \quad \text{any} \quad j.$$

The influence matrix accounts for the repair transitions that are influenced by the other agents:

$$\mathcal{I}^{X_j} = \begin{bmatrix} 0 & 0 \\ \mu_j \prod_{i<j} \pi_u^{X_i} & -\mu_j \prod_{i<j} \pi_u^{X_i} \end{bmatrix}.$$

And for each MA:

$$\mathrm{K}^{X_j} = \mathbf{Q}^{X_j} + \mathcal{I}^{X_j}.$$

In this very simple case, the influence matrix can be computed sequentially from $\mathcal{I}^{X_j}$ to $\mathcal{I}^{X_{j+1}}$. In more complex repair schemes [27, 28], the computation of the influence matrix requires a fixed-point iteration.

13.3.2 Message Passing Model

In the message passing model, the influence among MAs is represented by the exchange of relational entities, called *messages*, that are emitted by an MA and perceived by the other ones modifying their stochastic dynamics. The interaction among agents is ruled by a *perception function* that captures the sending and receiving aptitude of the involved MAs and is a function of their geographical location and of the features of the traversed media. MAs may belong to different classes with different local behaviours and interaction capabilities, and messages may belong to different types where each type induces a different effect on the interaction mechanism. The perception function describes how a message of a given type emitted by an MA of a given class in a given position in the space is perceived by an MA of a given class in a different position.

In particular, the message passing model considers a set of different messages $\mathcal{M}$, and a set $\mathcal{U} = \{u_1(\cdot)\ldots u_{|\mathcal{M}|}(\cdot)\}$ of $|\mathcal{M}|$ perception functions, one for each message type.

This model does not allow birth or death of agents ($\mathrm{b}_c(t, \mathbf{v}, [\Pi_\mathcal{V}]) = 0$ and $\mathbf{D}_c(t, \mathbf{v}, [\Pi_\mathcal{V}]) = 0$). Matrix $\mathbf{Q}_c(\mathbf{v})$ is the $n_c \times n_c$ infinitesimal generator matrix of the CTMC that describes the local behaviour of a class c agent in position $\mathbf{v}$, and corresponds to the local transition kernel of the general model. In order to consider the possibility to send messages without changing state, self-jump transitions are explicitly included in the model. In particular, the n_c component vector $\mathbf{\Lambda}_c(\mathbf{v}) = |\lambda_j^{\{c\}}(\mathbf{v})|$ represents the rates of *self-jumps* for a class c agent in position $\mathbf{v}$, i.e. the rates at which the CTMC reenters the same state.

The induced transition kernel $\mathcal{I}_c(t, \mathbf{v}, [\Pi_\mathcal{V}])$ is instead built starting from several parameters that characterize the way in which messages are sent and received. In particular, $\mathbf{G}_c(\mathbf{v}, \mu) = |g_{ij}^{\{c\}}(\mathbf{v}, \mu)|$ is a $n_c \times n_c$ matrix describing the probability that an agent of class c in position $\mathbf{v}$ generates a message of type $\mu \in 1\ldots|\mathcal{M}|$ during a jump from state i to state j, and $\mathbf{A}_c(\mathbf{v}, \mu) = |a_{ij}^{\{c\}}(\mathbf{v}, \mu)|$ is a $n_c \times n_c$ matrix, that describes the action activated upon acceptance of a type μ message for an agent of class c in position $\mathbf{v}$.

The perception function $u_\mu(\mathbf{v}, c, i, \mathbf{v}', c', i') \in [0, +\infty)$ represents the aptitude with which an agent of class c, in position $\mathbf{v}$, and in state i, perceives a message of type μ generated by an agent of class c' in position v' in state i'.

From the previous terms, we can define $\beta_j^{\{c\}}(\mathbf{v}, \mu)$ as the total rate at which messages of type μ are generated by an agent of class c in state j and in position $\mathbf{v}$:

$$\beta_j^{\{c\}}(\mathbf{v}, \mu) = \underbrace{\lambda_j^{\{c\}}(\mathbf{v}) g_{jj}^{\{c\}}(\mathbf{v}, \mu)}_{ⓐ} + \underbrace{\sum_{k \neq j} q_{jk}^{\{c\}}(\mathbf{v}) g_{jk}^{\{c\}}(\mathbf{v}, \mu)}_{ⓑ}, \tag{13.7}$$

where the first term (a) in the r.h.s is the contribution of the messages of type μ emitted during a self-loop from state j and the second term (b) is the contribution of messages of type μ emitted during a transition from state j to any state $k(\neq j)$.

Then we define $\gamma_{ii}^{\{c\}}(t, \mathbf{v}, [\Pi_\mathcal{V}], \mu)$ as the total rate at which messages of type μ coming from the whole volume V are perceived by an agent of class c, in state i, in position $\mathbf{v}$, at time t:

$$\gamma_{ii}^{\{c\}}(t, \mathbf{v}, [\Pi_\mathcal{V}], \mu) = \int\limits_{\substack{\mathbf{v}' \in \mathcal{V} \\ \mathbf{v}' \neq \mathbf{v}}} \sum_{c'=1}^{C} \sum_{j=1}^{n_{c'}} u_\mu(\mathbf{v}, c, i, \mathbf{v}', c', j) \beta_j^{\{c'\}}(\mathbf{v}', \mu) \pi_j^{\{c'\}}(t, \mathbf{v}') d\mathbf{v}'. \tag{13.8}$$

The term $u_\mu(\mathbf{v}, c, i, \mathbf{v}', c', j)\beta_j^{\{c'\}}(\mathbf{v}', \mu)\pi_j^{\{c'\}}(t, \mathbf{v}')$ in Eq. (13.8) is the rate of type μ messages received by class c agent in state i in position $\mathbf{v}$, coming from a class $c\prime$ agent in position $\mathbf{v}'$ in state j, at time t. Since this term depends on the current density of agents in other locations of the model $\pi_j^{\{c'\}}(t, \mathbf{v}')$, the value of $\gamma_{ii}^c(t, \mathbf{v}, [\Pi_\mathcal{V}], \mu)$ depends also on the total state of the model $[\Pi_\mathcal{V}]$. The total rate $\gamma_{ii}^c(t, \mathbf{v}, [\Pi_\mathcal{V}], \mu)$ is obtained integrating the contributions coming from all the states and all the agent classes, and over the entire area $\mathcal{V}$. Note that in discrete space models, the integral is replaced by a summation. We collect the rates of Eq. (13.8) in a diagonal matrix $\mathbf{\Gamma}_c(t, \mathbf{v}, [\Pi_\mathcal{V}], \mu) = \mathrm{diag}(\gamma_{ii}^{\{c\}}(t, \mathbf{v}, [\Pi_\mathcal{V}], \mu))$. This matrix can be used to compute the induced transitions kernel $\mathcal{I}_c(t, \mathbf{v}, [\Pi_\mathcal{V}])$:

$$\mathcal{I}_c(t, \mathbf{v}, [\Pi_\mathcal{V}]) = \sum_{\mu \in \mathcal{M}} \mathbf{\Gamma}^{\{c\}}(t, \mathbf{v}, [\Pi_\mathcal{V}], \mu)\Big(\mathbf{A}^{\{c\}}(\mathbf{v}, \mu) - \mathbf{I}\Big). \tag{13.9}$$

In Eq. (13.9) the acceptance matrix $\mathbf{A}^{\{c\}}(\mathbf{v}, \mu)$ is used to decide whether message μ received by an agent in state i will cause a transition (with probability $1 - a_{ii}^{\{c\}}(\mathbf{v}, \mu)$), and to which state j its reception will lead ($a_{ij}^{\{c\}}(\mathbf{v}, \mu)$).

The message passing model has been applied in the literature to several case studies. For example, in the field of Wireless Sensor Networks, it has been used in [17] to study on-off policies, and in [3] to evaluate swarm intelligence based routing algorithms. In [6] messages are used to study the propagation of earthquakes, while in [9] they are used to study the propagation of fire. Finally, in [8] agent are used at two levels: to model both a physical phenomenon (fire propagation in forest) and a WSN monitoring infrastructure.

13.3.3 Spatial Density Dependent Communications

In the message passing model presented in Sect. 13.3.2, the value of the perception function depends on the type of message exchanged between two MAs, a sender and a receiver, and on their properties (location, class, and current state), as formally described by the notation $u_\mu(\mathbf{v}, c, i, \mathbf{v}', c', i')$. However, this dependency causes a one-to-one relationship that restricts the expressiveness of the model since the influence matrix $\mathcal{I}_c(t, \mathbf{v}[\Pi_\mathcal{V}])$ would in principle allow to represent one-to-all interactions. An effective trade-off between these extremes is given by considering one-to-many relations, where several neighbours of the perceiver agent are taken into account. In the following, we will describe an example of such approach to model the *Double Bridge Experiment* [12, 16], a famous ant colony optimization (ACO) problem [13].

In the experiment, two bridges connect a nest of ants with a food source. Two scenarios are investigated: in the former the lengths of the bridges are equal, in the latter a bridge is shorter than the other one. The experiment shows that using *stigmergy*, a form of indirect communication through the environment, the ant colony is able to reach the food following the shortest path. Indeed, during the journey from the nest to the food and vice versa, ants release an amount of a chemical substance called *pheromone*. Ants perceive it and are conditioned to choose with greater probability a path marked with a strong concentration of pheromone. In presence of bridges of different length, the ants choosing the shortest one reach the food earlier than those choosing the longer path. Thus, the pheromone trail increases faster on the shorter bridge and further ants choose it to reach food. In the end, the ant colony converges to follow the shortest path.

The dynamic of the ant colony in this experiment can be described by a simple stochastic model [12, 16], where the probability of choosing a given branch is given by:

$$p_{is}(t) = \frac{(k+\varphi_{is}(t))^{\alpha}}{(k+\varphi_{is}(t))^{\alpha} + (k+\varphi_{il}(t))^{\alpha}}, \quad p_{il}(t) = 1 - p_{is}(t), \tag{13.10}$$

where $p_{is}(t)$ is the probability of choosing the shorter branch, and $\varphi_{is}(t)$ is the amount of pheromone on the shorter branch at a time t. The same values on the longer branch are given by $p_{il}(t)$ and $\varphi_{il}(t)$. The parameter k is needed to provide a non-null probability of choosing a path not yet marked by pheromone; the exponent α provides a non-linear behaviour. We can observe that in such case, the probability of choosing a specific direction depends on the spatial distribution of the pheromone concentrations.

The MAM of the *Double Bridge Experiment* proposed in [4] represents ants as messages, and locations that ants traverse by MAs. More formally, agents are deployed on a geographical space $\mathcal{V}$ structured as an undirected graph $G = (\mathcal{V}, E)$ with $\mathcal{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots \mathbf{v}_N\}$ the set of locations where the MAs reside, and E the edges of the graph. Figure 13.2 shows the graphs for the two scenarios of the experiments, where nodes labelled n and f represent the location of nest and source food, respectively. Messages passing from a location to another are depicted as little arrows with a label indicating the direction of the ants. Label m_{fw} for ants directed to the source food, m_{bw} for ants coming back to the nest. The number of hops from node n to node f represents the length of the path, thus Fig. 13.2a depicts a scenario with equal branches, whereas Fig. 13.2b the different branches one.

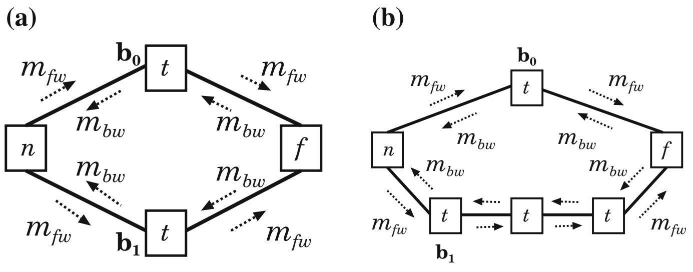

Fig. 13.2 Graph used to model the experiment scenarios. **a** Equal branches, **b** two different branches

The amount of pheromone is discretized in a finite number of levels and the MA in location $\mathbf{v}$ codes in its state space the current level of pheromone in that location. The arrival of ants causes the increment of the pheromone and is represented in the model by the reception of messages and consequent transition from a lower to an higher level state of the MA in location $\mathbf{v}$. In the model the birth and death of MAs are not considered (i.e. $\mathbf{b}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) = 0$ and $\mathbf{D}_c(t, \mathbf{v}, [\Pi_{\mathcal{V}}]) = 0$). Moreover, since the flow of messages represent the moving ants across a set of fixed locations, the term $\boldsymbol{\omega}(t, \mathbf{v}, [\Pi_{\mathcal{V}}])$ is neglected too. Thus, the general Eq. (13.1) became

$$\frac{\partial \boldsymbol{\pi}(t,\mathbf{v})}{\partial t} = \boldsymbol{\pi}(t,\mathbf{v}) \cdot \mathbf{K}(t,\mathbf{v},\boldsymbol{\pi}_{\mathcal{V}}(t)), \tag{13.11}$$

where the transition kernel $\mathbf{K}(t,\mathbf{v},\boldsymbol{\pi}_{\mathcal{V}}(t))$ depends on the ensemble of the probability distribution of all agents at current time t only. In particular, it can be computed as described in Sect. 13.3.2 where the perception function $u_\mu(\cdot)$ with $\mu \in \{m_{fw}, m_{bw}\}$ is defined as

$$u_\mu(\mathbf{v},\mathbf{v}',t) = \frac{(k + E[\boldsymbol{\pi}(t,\mathbf{v})])^\alpha}{\sum_{v'' \in Next^\mu(\mathbf{v}')} (k + E[\boldsymbol{\pi}(t,\mathbf{v}'')])^\alpha}. \tag{13.12}$$

Parameters k and α are the same of Eq. (13.10), $E[\boldsymbol{\pi}(t,\mathbf{v})]$ gives the mean value of the concentration of pheromone at a time t in position $\mathbf{v}$. The function $Next(\mathbf{v}')$ gives the set of elements $\{\mathbf{v}''\}$ such that the agent in position $\mathbf{v}''$ perceives a message emitted by the agent of in position $\mathbf{v}'$. Note that, even if the perception function is defined on a pair of sender and receiver MAs as in Sect. 13.3.2, its value depends on the properties of a set of agents that in this case are the neighbours of the sender MA.

13.3.4 Agent Motion Models

In Agent Motion Models (AMMs), the MAs interact each others by exchanging messages, and move across a continuous geographical space $\mathcal{V} \subseteq \mathbb{R}^k$ with $k \leq 3$. For simplicity in the following we apply some restrictions, in particular: (i) we focus on single-class AMMs, thus dropping the subscript c and superscript $\{c\}$ in the equations, (ii) we consider a one-dimensional geographical space, i.e. a straight line, by setting $k = 1$ and (iii) we exclude "warp" phenomena by setting both terms $\mathbf{m}^{in}(t, \mathbf{v}, [\boldsymbol{\pi}_{\mathcal{V}}]) = 0$ and $\mathbf{M}^{out}(t, \mathbf{v}, [\boldsymbol{\pi}_{\mathcal{V}}]) = 0$.

With such assumptions, Eq. (13.1) became

$$\frac{\partial \boldsymbol{\pi}(t,x)}{\partial t} + \frac{\partial(\boldsymbol{\omega}(t,x,[\boldsymbol{\pi}_{\mathcal{V}}]) \cdot \boldsymbol{\pi}_c(t,x)^T)}{\partial x} = \mathbf{b}(t,v,[\boldsymbol{\pi}_{\mathcal{V}}]) + \boldsymbol{\pi}(t,x) \cdot \mathbf{K}(t,x,[\boldsymbol{\pi}_{\mathcal{V}}]), \tag{13.13}$$

where the transition kernels $\mathbf{K}(t, x, [\boldsymbol{\pi}_{\mathcal{V}}])$ is defined as:

$$\begin{aligned}\mathbf{K}(t, x, [\boldsymbol{\pi}_{\mathcal{V}}]) = \mathbf{Q}(t, x) + \mathcal{I}(t, x, [\boldsymbol{\pi}_{\mathcal{V}}]) \\ + \mathbf{D}(t, x, [\boldsymbol{\pi}_{\mathcal{V}}]).\end{aligned} \tag{13.14}$$

Equation (13.13) can be solved applying *upwind semi-discretization* [22]. The partial differential equation is first discretized over the space, and then solved with respect to the time. The *upwind* technique makes a first-order approximation of the spatial derivative with respect to the direction of movements. As a result, we obtain a system of ordinary differential equations that can be solved using standard methods like Euler or Runge–Kutta.

AMMs can be exploited to study the behaviour of Intelligent Transportation Systems (ITS) [26], where information and communication technologies are applied to the design, analysis, monitoring and control of transportation systems, with particular emphasis on road networks. Transportation systems can involve multiple entities, such as humans, vehicles and the physical infrastructure, interacting each others. Several aspects of transportation systems are uncertain and nonlinear. They are usually large scale and are always geographically distributed. For all such reasons, AMMs are suitable to describe and study performance of the ITS in terms of road traffic congestion, safety and so on. An AMM was first applied to ITS for the quantitative risk analysis of collision of vehicles in a road tunnel [7]. The model describes the behaviour of a flow of vehicles capable to sense their proximity to other cars or to unexpected obstacles. According to such informations, the vehicles automatically adapt their speed in order to preserve a prescribed safety distance, or even brake to avoid collisions. In the model, MAs code in their space-state the cruise speed of vehicles. Proximity messages are periodically exchanged among MAs and influence such speeds as described by the $\mathcal{I}(t, x, [\boldsymbol{\pi}_{\mathcal{V}}])$ term. The perception function rules the reaction of the vehicle to the proximity with other cars or obstacles. Birth term $\mathbf{b}(t, x, [\boldsymbol{\pi}_{\mathcal{V}}])$ and death term $\mathbf{D}(t, x, [\boldsymbol{\pi}_{\mathcal{V}}])$ account for the entrance and exit of vehicle in the tunnel, respectively. The results provided by the model were used to compute the values of the safety distance and the maximum allowed speed that minimize the probability of collisions inside the tunnel.

13.3.5 Population Models

Population Markovian Agents (PMAs) are Markovian Agents models where agents can move to other locations, can increase in number or decrease (either spontaneously or induced by other agents), or they can multiply during the transitions.

To simplify the presentation, we will consider a single class of agents, and we will drop subscript c or superscript $\{c\}$ from the equations. We will also focus on a discrete space model $\mathcal{V} = \{\mathbf{v_1}, \mathbf{v_2}, \ldots \mathbf{v_N}\}$. A PMA describes the evolution of a single agent, and it is defined by a tuple $(\mathcal{Q}, b, d, R, \eta, N)$. The first three components define the common Markovian Agent behaviour, while the last three account for the population evolution of the system. $\mathcal{Q} = (\widetilde{Q}(\mathbf{v}), \widehat{Q}^{[k]}(\mathbf{v}, \mathbf{v}'))$, are the *transition rate matrices*. $\tilde{q}_{ij}(\mathbf{v})$ is the rate at which an agent in location $\mathbf{v}$ jumps from state i to j due to local activities. $\widehat{Q}_{ij}^{[k]}(\mathbf{v}', \mathbf{v})$ accounts for transitions caused by inductions, and represents the rate at which an agent in state k in position $\mathbf{v}'$ induces jumps from state i to state j in position $\mathbf{v}$. In Eq. (13.4) we have

$$\mathbf{Q}_c(t, \mathbf{v}) = \widetilde{Q}(\mathbf{v}), \quad \mathcal{I}_c(t, \mathbf{v}, [\boldsymbol{\pi}_\mathcal{V}]) = \sum_{\mathbf{v}' \in \mathcal{V}} \sum_k \widehat{Q}^{[k]}(\mathbf{v}', \mathbf{v}) \boldsymbol{\pi}_k(t, \mathbf{v}'). \tag{13.15}$$

Component $b = (\tilde{b}(l), \hat{b}^{[k]}(\mathbf{v}, \mathbf{v}'))$, where $\tilde{b}(\mathbf{v}) = |\tilde{b}_i(\mathbf{v})|$ and $\hat{b}^{[k]}(\mathbf{v}) = |\hat{b}_i^{[k]}(\mathbf{v}', \mathbf{v})|$, are respectively the spontaneous and induced *births vectors*. In particular, $\tilde{b}_i(\mathbf{v})$ represents the rate at which agents are created in state i at location $\mathbf{v}$, and $\hat{b}_i^{[k]}(\mathbf{v}', \mathbf{v})$ is the rate at which an agent in state k in position $\mathbf{v}'$ induces birth of agents in state i at location $\mathbf{v}$. In a similar way, $d = (\tilde{d}(\mathbf{v}), \hat{d}^{[k]}(\mathbf{v}, \mathbf{v}'))$, where $\tilde{d}(\mathbf{v}) = |\tilde{d}_i(\mathbf{v})|$ and $\hat{d}^{[k]}(\mathbf{v}', \mathbf{v}) = |\hat{d}_i^{[k]}(\mathbf{v}', \mathbf{v})|$ are respectively the spontaneous and induced *death vectors*, where $\tilde{d}_i(\mathbf{v})$ represents the rate at which agents are destroyed in state i at location $\mathbf{v}$, and $\hat{d}_i^{[k]}(\mathbf{v}', \mathbf{v})$ is the rate at which an agent in state k and position $\mathbf{v}'$ induces decrease in the number of agents in state i at location $\mathbf{v}$. In Eqs. (13.3) and (13.4), we have that the birth and death kernels can be defined as

$$\mathbf{b}_c(t, \mathbf{v}, [\boldsymbol{\pi}_\mathcal{V}]) = \tilde{b}(\mathbf{v}) + \sum_{\mathbf{v}' \in \mathcal{V}} \sum_k \hat{b}^{[\mathrm{k}]}(\mathbf{v}', \mathbf{v}) \boldsymbol{\pi}_k(t, \mathbf{v}'), \tag{13.16}$$

$$\mathbf{D}_c(t, \mathbf{v}, [\boldsymbol{\pi}_\mathcal{V}]) = \mathrm{diag}\left(\tilde{d}(\mathbf{v}) + \sum_{\mathbf{v}' \in \mathcal{V}} \sum_k \hat{d}^{[k]}(\mathbf{v}', \mathbf{v}) \boldsymbol{\pi}_k(t, \mathbf{v}') \right). \tag{13.17}$$

The peculiarity of PMAs are the set of *reactions* $R = \{r_1, \ldots, r_{|R|}\}$ that allows agents to move, duplicate or merge, either in the same or in neighbour locations. Reactions are characterized by a *reaction vector* $\eta(\mathbf{v}, [\boldsymbol{\pi}_\mathcal{V}]) = |\eta_h(\mathbf{v}, [\boldsymbol{\pi}_\mathcal{V}])|$, with $\mathbf{v} \in \mathcal{V}$. In particular, $\eta_h(\mathbf{v}, [\boldsymbol{\pi}_\mathcal{V}])$ represents the speed at which reaction r_h occurs in location $\mathbf{v}$. Since reaction rates can have very complex expressions, they can be the functions of the complete state of the model. The effects of reactions are described by $\mathcal{N} = (\mathcal{N}^+(\mathbf{v}, \mathbf{v}''), \mathcal{N}^-(\mathbf{v}, \mathbf{v}''))$ where $\mathcal{N}^+(\mathbf{v}, \mathbf{v}'') = |n_{i,k}^+(\mathbf{v}, \mathbf{v}'')|$ and $\mathcal{N}^-(\mathbf{v}, \mathbf{v}'') = |n_{i,k}^-(\mathbf{v}, \mathbf{v}'')|$ are $n_c \times |R|$ matrices that describe the changes on the number of agents due to the occurrence of the reaction $r_k \in R$. The value of $n_{i,k}^+(\mathbf{v}, \mathbf{v}'')$ represents the number of agents that are added to state i in location $\mathbf{v}''$ when reaction r_k takes place, while $n_{i,k}^-(\mathbf{v}, \mathbf{v}'')$ accounts for the agents that are removed. The formalization of terms η and $\mathcal{N}$ is quite similar to one used in computational system biology (see [14]). Reactions can also multiply or divide the number of agents in a given location. In this case, both $n_{j,e}^-(\mathbf{v}, \mathbf{v}'') > 0$ and

$n^+_{j,e}(\mathbf{v},\mathbf{v}'') > 0$ for the same reaction r_e and state j. The definitions of η and $\mathcal{N}$ are expressive enough to model several different features: duplication or decimation of agents in a state, movement of agents (deterministic or probabilistic) to other locations, duplication or decimation combined with state jump and location movement. Interested readers can find more details in [11]. Reactions can be implemented in Eq. (13.1) by defining the input and output kernels as follows:

$$\mathbf{m}^{in}_c(t,\mathbf{v},[\boldsymbol{\pi}_{\mathcal{V}}]) = \sum_{\mathbf{v}''\in L} \eta(\mathbf{v}'',[\boldsymbol{\pi}_{\mathcal{V}}])(N^+(\mathbf{v}'',\mathbf{v}))^T, \tag{13.18}$$

$$\mathbf{M}^{out}_c(t,\mathbf{v},[\boldsymbol{\pi}_{\mathcal{V}}]) = \mathrm{diag}(\boldsymbol{\pi}(t,\mathbf{v}))^{-1} \sum_{\mathbf{v}''\in L} \eta(\mathbf{v},[\boldsymbol{\pi}_{\mathcal{V}}])(N^-(\mathbf{v},\mathbf{v}''))^T, \tag{13.19}$$

where $(\mathcal{N})^T$ denotes the transpose of matrix $\mathcal{N}$. Equation (13.18) considers agents that are entering a location $\mathbf{v}$, coming from a reaction that takes place in a location $\mathbf{v}''$. Equation (13.19) accounts for the change in the number of agents occurring in one state due to elements removed by the reactions. In particular, it accounts for all the agents that are leaving location $\mathbf{v}$ directed to a reaction happening in location $\mathbf{v}''$. Since the rate at which reactions occurs already accounts for the density of the agents, in order to allow the inclusion in the output kernel of Eq. (13.4), we must normalize the effect by multiplying on the left by $\mathrm{diag}(\boldsymbol{\pi}(t,\mathbf{v}))^{-1}$.

The definition of $\eta(\mathbf{v},[\boldsymbol{\pi}_{\mathcal{V}}])$ must depend on the total state of the system $[\boldsymbol{\pi}_{\mathcal{V}}]$, and it must be correctly defined to prevent reactions to happen when there are not enough agents in the involved states: an improperly defined function $\eta(\mathbf{v},[\boldsymbol{\pi}_{\mathcal{V}}])$ can lead to negative counts in the number of agents. Determining the condition for which a function $\eta(\mathbf{v},[\boldsymbol{\pi}_{\mathcal{V}}])$ does not lead to negative counts, is an important topic that will be investigated in future woks: here we will limit ourselves to consider functions coming from system biology, that are known to correctly behave as reaction rates. In particular we refer to the *Law of Mass Action* which was studied by Waage and Guldberg in 1864. Such law tells that the reaction rate is proportional to the probability of a collision of the reactants, that in turn is proportional to the concentration of reactants (agents in our case), elevated to the multiplicity required to start the reaction:

$$\eta_h(\mathbf{v},[\boldsymbol{\pi}_{\mathcal{V}}]) = \gamma_h \prod_{j:\mathcal{V}^-(\mathbf{v},j)\neq\emptyset} \left(\sum_{\mathbf{v}''\in\mathcal{V}^-(\mathbf{v},j)} \boldsymbol{\pi}_j(t,\mathbf{v}') \right)^{\sum_{\mathbf{v}''\in\mathcal{V}^-(\mathbf{v},j)} (n^-_{j,h}\mathbf{v},\mathbf{v}'')}. \tag{13.20}$$

Here γ_h represents the speed at which the reaction occurs. Note that since the *source* agents of reaction r_h in location $\mathbf{v}$ can arrive from any location $\mathbf{v}''$ such that $n^-_{j,h}(\mathbf{v},\mathbf{v}'') > 0$, (here denoted with $\mathcal{V}^-(\mathbf{v},j) = \{\mathbf{v}' \in \mathcal{V} : n^-_{j,h}(\mathbf{v},\mathbf{v}') > 0\}$), the total count of agents required to engage a reaction must be computed with the sum $\sum_{\mathbf{v}''\in\mathcal{V}^-(\mathbf{v},j)} n^-_{j,h}(\mathbf{v},\mathbf{v}'')$. For the same reason, the number of agents involved in the reaction, must be computed with the sum over all the possible input locations, i.e. $\sum_{\mathbf{v}' \in \mathcal{V}^-(\mathbf{v},j)} \boldsymbol{\pi}_j(t, \mathbf{v}')$.

PMAs are applied in [11] to analyze the cancer evolution, and to comprehend the mechanisms underlying the Cancer Stem Cells hierarchy whose characterization is crucial in the study of tumour progression. Current evidence indicates that many cancers arise from a small population of cells named Cancer Stem Cells (CSCs), which have undergone malignant transformation driven by frequent genetic mutations. The application of the PMAs to describe the behaviour of such cell populations allowed the scientists to observe their proliferation through the host tissue. The model is exploited to consider movement of cell populations in a bi-dimensional space, and it is used to derive the system evolution.

Recent studies in cancer biology have led to a new perspective in tumour progression, known as the CSC theory. It states that the growth and evolution of many cancers are driven by a small population of cells named CSC, and that CSC-based tumours are hierarchically structured, and characterized by different subpopulations of cells: CSCs, Progenitor Cells (PCs) and Totally differentiated Cells (TCs). Moreover, such heterogeneity is considered the cause of the failure of many conventional therapies. It is hence fundamental to fully comprehend the CSC CSC dynamics to predict treatment response. The proposed PMA model describes the CSC-based tumour growth and it is able to reproduce the overall dynamics among cell subpopulations during tumour progression. Using a derivation similar to the Generalized Mass Action law, the reactions describing the biological model were translated into a system of Ordinary Differential Equations (ODEs) able to take into account the possibility of reactions to expand in neighbour cells.

13.3.6 Abstract Space Models

When space $\mathcal{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots\}$ is discrete, it can be used not only to describe graph-based routes, but also to represent more abstract configurations. For example, in [5] locations are used to model different data centres of a geographically distributed cloud infrastructure. In that case $\mathcal{V} = \{dc_1, dc_2, \ldots\}$ locations are used to model the different data centres composing the infrastructure. Dependency on the location is used to assign different capabilities in terms of computational nodes that are able to run Virtual Machines (VMs), and disk infrastructure capable of saving data as Storage Blocks (SBs).

Agents classes $1 \leq c \leq C$ represents applications running in the data centre: the state of the agent defines the resource usage of each application in the infrastructure. The number of applications for each class c running at data centre dc_j is encoded into the agent density function $\rho_c(t, dc_j)$. The average performance of the nodes of the data centre is encoded in the transition function $\widetilde{\mathbf{K}}_c([\boldsymbol{\pi}_{\mathcal{V}}])$ for each application class. In this case, in particular, the local transition kernel $\mathbf{Q}_c(t, \mathbf{v}) = 0$ since the speed at which application acquires and releases resources depends on the entire state of the data centre, and $\mathcal{I}_c(t, \mathbf{v}) = \widetilde{\mathbf{K}}_c([\boldsymbol{\pi}_{\mathcal{V}}])$.

If we consider a fixed number of applications, the birth term and death term can be set to $\mathbf{b}_c(t, \mathbf{v}, [\boldsymbol{\pi}_{\mathcal{V}}]) = 0$ and $\mathbf{D}_c(t, \mathbf{v}, [\boldsymbol{\pi}_{\mathcal{V}}]) = 0$, otherwise they can be used to model the starting and stopping of applications in presence of fluctuations in the workload. The motion terms $\mathbf{M}_c^{out}(t, \mathbf{v}, [\boldsymbol{\pi}_{\mathcal{V}}])$ and $\mathbf{m}_c^{in}(t, \mathbf{v}, [\boldsymbol{\pi}_{\mathcal{V}}])$ can be used to model application migration from one data centre to the other when supporting load-balancing applications that works at the complete geographical infrastructure level.

Abstract space models have also been used in [19] to model cell hand-over and technology switch in wireless network models, where user agents can connect to either a 4G LTE network or to a WiFi access point depending on the current load.

13.3.7 Delay Models

The ability of agents that perceive the state of the model even at previous time instants $[\boldsymbol{\pi}_{\mathcal{V}}]$, can be used to add random or deterministic delays in the delivery of messages to the model presented in Sect. 13.3.2. In particular, let us assume that the delivery of message μ sent from a class c agent in position $\mathbf{v}$ to a class c' agent in position $\mathbf{v}'$ during a transition from state i to state j required a random time τ distributed according to a positive continuous distribution $Y_\mu(\tau|\mathbf{v}, c, i, \mathbf{v}', c', j)$. The time delay in messages can be considered by replacing the definition of $\gamma_{ii}^{\{c\}}(t, \mathbf{v}, [\boldsymbol{\pi}_{\mathcal{V}}], \mu)$ previously given in Eq. (13.8) with:

$$\gamma_{ii}^{\{c\}}(t, \mathbf{v}, [\boldsymbol{\pi}_{\mathcal{V}}], \mu) = \int\limits_{\substack{\mathbf{v}' \in \mathcal{V} \\ \mathbf{v}' \neq \mathbf{v}}} \sum_{c'=1}^{C} \sum_{j=1}^{n_{c'}} \beta_j^{\{c'\}}(\mathbf{v}', \mu) u_\mu(\mathbf{v}, c, i, \mathbf{v}', c', j) \int\limits_0^t \boldsymbol{\pi}_j^{\{c'\}}(t - \tau, \mathbf{v}') \mathrm{d}Y_\mu(\tau|\mathbf{v}, c, i, \mathbf{v}', c', j) d\mathbf{v}'. \tag{13.21}$$

In this new definition, the time integral over τ accounts for the state of the system at time $t - \tau$, weighted by the probability that the delay distribution $Y_\mu(\tau|v, c, i, \mathbf{v}', c', j)$ is equal to τ. This definition makes Eq. (13.1), which normally is either an ordinary differential equation (if motion over a continuous space is not considered) or a partial differential equation, a *delay differential equation*. Although the techniques for the solution of delay differential equation are very solid, they can be computationally expensive and reduce the size of models that could be fruitfully analyzed. From a theoretical point of view, however, this result enhances the possibility of MA modelling: a full investigation of the advantages of using delay differential equation models, and the definition of the use cases when this can provide good results is ongoing research work.

13.4 Conclusions

In this work, we have presented a unified theory that can be used to study various types of Markovian Agent models. In particular, we have shown that a single equation can describe all the peculiarities of agents: local transitions, induced behaviours, motion, increase and decrease in population. Markovian Agents have been used to study a variety of different systems, both physical and IT related: in this work we have provided some hints on how the formalism has been used, and given links to the literature where such cases have been discussed.

Even if the theory is already solid, many features and limitations of Markovian Agents still need to be investigated: future works will focus on improving solution techniques, providing tools to simplify the use of MA models, defining best-practices and study the scalability and accuracy of MA models.

References

1. Ball F, Milne R, Tame I, Yeo G (1997) Superposition of interacting aggregated continuous-time Markov chains. Adv Appl Probab 29:56–91
2. Bobbio A, Gribaudo M, Telek M (2008) Analysis of large scale interacting systems by mean field method. In: 5th international conference on quantitative evaluation of systems—QEST2008. St. Malo
3. Bruneo D, Scarpa M, Bobbio A, Cerotti D, Gribaudo M (2012) Markovian agent modeling swarm intelligence algorithms in wireless sensor networks. Perform Eval 69:135–149
4. Bruneo D, Scarpa M, Bobbio A, Cerotti D, Gribaudo M (2015) An intelligent swarm of markovian agents. In: Springer handbook of computational intelligence, pp 1345–1359
5. Castiglione A, Gribaudo M, Iacono M, Palmieri F (2015) Modeling performances of concurrent big data applications. Softw Pract Exp 45(8):1127–1144. doi:10.1002/spe.2269
6. Cerotti D, Gribaudo M, Bobbio A (2009) Disaster propagation in inhomogeneous media via markovian agents. In: Critical information infrastructure security, vol 5508, pp 328–335. Springer LNCS (2009)
7. Cerotti D, Gribaudo M, Bobbio A (2009) Presenting dynamic markovian agents with a road tunnel application. In: 17th annual meeting of the IEEE/ACM international symposium on modelling, analysis and simulation of computer and telecommunication systems, MASCOTS 2009, September 21–23, 2009, South Kensington Campus, Imperial College London, pp 1–4
8. Cerotti D, Gribaudo M, Bobbio A (2014) Markovian agents models for wireless sensor networks deployed in environmental protection. Rel Eng Syst Saf 130:149–158
9. Cerotti D, Gribaudo M, Bobbio A, Calafate CMT, Manzoni P (2010) A markovian agent model for fire propagation in outdoor environments. In: Computer Performance Engineering—7th European Performance Engineering Workshop, EPEW 2010, Bertinoro, Italy, September 23–24, 2010. Proceedings, pp 131–146
10. Ciardo G, Muppala J, Trivedi K (1991) On the solution of GSPN reward models. Perform Eval 12:237–253
11. Cordero F, Fornari C, Gribaudo M, Manini D (2014) Markovian agents population models to study cancer evolution. In: Analytical and stochastic modelling techniques and applications—21st international conference, ASMTA 2014, Budapest, Hungary, June 30–July 2, 2014. Proceedings, pp 16–32

12. Deneubourg Aron S, Goss S, Pasteels JM (1990) The self-organizing exploratory pattern of the argentine ant. J Insect Behav 3(2):159–168. doi:10.1007/BF01417909
13. Dorigo M, Stützle T (2004) Ant colony optimization. MIT Press
14. Edwards J, Palsson B (1998) How will bioinformatics influence metabolic engineering? Biotechnol Bioeng 58(2–3):162–169
15. Gilmore S, Hillston J, Kloul L, Ribaudo M (2003) PEPA nets: a structured performance modelling formalism. Perform Eval 54(2):79–104. http://www.sciencedirect.com/science/article/pii/S0166531603000695
16. Goss S, Aron S, Deneubourg J, Pasteels J (1989) Self-organized shortcuts in the Argentine ant. Naturwissenschaften 76(12):579–581. doi:10.1007/BF00462870
17. Gribaudo M, Cerotti D, Bobbio A (2008) Analysis of on-off policies in sensor networks using interacting Markovian agents. In: 4th international work sensor networks and systems for pervasive computing—PerSens 2008, pp 300–305
18. Gribaudo M, Chiasserini CF, Gaeta R, Garetto M, Manini D, Sereno M (2005) A spatial fluid-based framework to analyze large-scale wireless sensor networks. In: IEEE international conference on dependable systems and networks, DSN2002
19. Gribaudo M, Manini D, Chiasserini C (2013) Studying mobile internet technologies with agent based mean-field models. In: Analytical and stochastic modelling techniques and applications—20th international conference, ASMTA 2013, Ghent, Belgium, July 8–10, 2013. Proceedings, pp 112–126
20. Guenther, M.C., Bradley, J.T.: Higher moment analysis of a spatial stochastic process algebra. In: Computer Performance Engineering, pp. 87–101. Springer - LNCS, Vol 6977, Springer (2011)
21. Hillston J (2005) Fluid flow approximation of PEPA models. In: 2nd international conference on quantitative evaluation of systems—QEST, pp 33–43
22. Horton G, Kulkarni VG, Nicol DM, Trivedi KS (1998) Fluid stochastic petri nets: theory, application, and solution techniques. Eur J Oper Res 105(1):184–201
23. Kaâniche M, Lollini P, Bondavalli A, Kanoun K (2012) Modeling the resilience of large and evolving systems. CoRR abs/1211.5738
24. Kurtz T (1981) Approximation of population processes. Soc Ind Appl Math. http://epubs.siam.org/doi/abs/10.1137/1.9781611970333
25. Plateau B, Atif K (1991) Stochastic automata network for modeling parallel systems. IEEE Trans Softw Eng 17:1093–1108
26. Sheng-hai A, Byung-Hyug L, Dong-Ryeol S (2011) A survey of intelligent transportation systems. In: 2011 Third international conference on computational intelligence, communication systems and networks (CICSyN), pp 332–337
27. Sukhwani H, Bobbio A, Trivedi K (2015) Largeness avoidance in availability modeling using hierarchical and fixed-point iterative techniques. Int J Perform Eng 11(4):305–319
28. Tomek L, Trivedi K (1991) Fixed point iteration in availability modeling. In: Cin M, Hohl W (eds) Fault-Tolerant Computing Systems, *Informatik-Fachberichte*, vol 283. Springer, Berlin, pp 229–240
29. Trivedi KS (2002) Probability and statistics with reliability, queuing and computer science applications. Wiley, Chichester
30. Trivedi KS, Ma X, Dharmaraja S (2003) Performability modelling of wireless communication systems. Int J Commun Syst 16(6):561–577. doi:10.1002/dac.605