

Efficiency bounds in computations on open multi-agent systems

Dissertation presented by
Charles MONNOYER DE GALLAND DE CARNIÈRES

for obtaining the Master's degree in
Mathematical Engineering

Supervisor(s)
Julien HENDRICKX

Reader(s)
Paolo FRASCA, Raphaël JUNGERS

Academic year 2017-2018

Abstract

Multi-agent systems are systems constituted of multiple independent and interactive entities acting towards some objective. Really widespread for their advantages in solving big problems in a decentralized way, it is getting needed to extend them to the case where arrivals and departures of agents happen. This case arises as the size of the systems grows, and through the study of some problems that naturally present such a property (such as computer failures in a network). Such systems are called Open Multi-Agent Systems, and represent a really young theory in need for development. A step in the understanding of such a theory consists in establishing algorithms and analyses for basic problems, that can be the base of more complex tasks. In this Master thesis, we intend to determine fundamental performance limitations on basic consensus problems in the framework of Open Multi-Agent Systems, under the form of bounds that can be used as quality criteria for algorithms solving these problems.

Key Words: multi-agent systems, open multi-agent systems, consensus problem, fundamental performance limitations.

Acknowledgements

First of all, I want to express my thankfulness to my supervisor Prof. Julien Hendrickx for his constant support throughout the realization of my master thesis. This work would never have been what it is without his help, his patience, and his continuous desire to make me improve. I really appreciated his interest in this work, and his involvement in guiding me with his considerable knowledge, that made possible to produce this final rendering of my work.

I also want to thank Balázs Gerencsér for his manifest interest for my work, and the help he provided for trying to solve one of the problems I faced in the realization of this master thesis.

I am also grateful to Prof. Raphaël Jungers and Prof. Paolo Frasca for the interest they shown for this work, and the time they agreed to spend on it.

Finally, I would like to thank my family and friends who supported me all along, Gilles for his help and proofreading anytime I asked him, Gwennaëlle for her constant advice and support, and Maud and Florian, for the unexpected help they provided me as the end of my work was approaching.

Contents

Abstract	1
Acknowledgements	2
Introduction	5
1 Average Consensus Problem	10
1.1 Problem statement	10
1.1.1 Assumptions on knowledge	10
1.1.2 Definition of the estimator and the error	11
1.2 Bound for two agents in the system	12
1.2.1 Computation	12
1.2.2 Observations	14
1.3 Ping model	15
1.3.1 Computation procedure	16
1.3.2 Results	17
1.3.3 Discussion	20
1.4 Development of a general method	21
1.4.1 New procedure	21
1.4.2 Discussion on the choice of the pdf	23
1.4.3 Results: Infection model	25
1.4.4 Discussion on the Infection model	29
1.5 A word on the assumptions	31
2 Mean Consensus Problem	34
2.1 Problem statement	34
2.1.1 Assumptions on knowledge	34
2.1.2 Definition of the estimator and the error	35
2.2 Definition of the bound	36
2.2.1 Description of the ideal algorithm	36
2.2.2 Expression for the bound	37
2.3 Simulations	39
2.3.1 Perspectives	39
2.3.2 Calibration of the simulations	40
2.3.3 Results	42
2.3.4 Discussion	46
Conclusion	48
Appendices	50

A	Local error for the Ping model with three agents	51
B	Local error for the Ping model with N agents	52
C	Bounds for several CDFs: Ping, Ping($t/2$) and 1	54
D	Computation of the Infection bound by hand	56

Introduction

Multi-Agent Systems: decentralizing information

We define a *Multi-Agent System* as a set of independent and interacting entities, generally acting towards some goal. Think for instance of drone formations independently flying in order to be at the right place at the right time to produce impressive figures. Or think for instance of groups of routers transmitting information from some point to another.

Many applications can be described using a multi-agent approach. However, dealing with Multi-Agent Systems is a challenging task: each agent being an independent entity, prediction and analysis can reveal to be tough, and rely on stochastic approaches to estimate some expected behavior. In return, it provides many advantages since the treatment of problems becomes collaborative and decentralized (*i.e.* it doesn't rely on a centralized unit monitoring the information). This explains why this approach has become more and more widespread in many fields.

By definition, solving a problem using a Multi-Agent Systems approach implies avoiding the centralization of information in one place. In such systems, each agent maintains its own approximation of the solution of the problem it attempts to solve, adaptive in time as the system evolves, instead of relying on some centralized units to provide it. This has the major impact to distribute the contribution on the information shared by the agents among them. This makes possible to work on big systems, as agents don't need to know the entire structure of the system they are part of. Furthermore, the absence of centralized units remove the risk of having a failure of the whole system if one of them fails: this risk is distributed among the agents too. Moreover, as agents maintain their estimation of the solution of the problem they try to solve, it is not required anymore to wait for the centralized units to provide it. On top of that, decentralized method is an opportunity to take advantage of the potential of smart recent devices through local interactions, while avoiding long-distance expensive communications.

Those advantages of working in a decentralized way are even more relevant with the increasing size of the systems. Indeed, as this size grows, the proportion of risks induced by centralized approaches are even growing too.

Many examples of Multi-Agent Systems can be listed, lying in many different types of applications. The coordination of drones or mobile robots such as UAV (Unmanned Aerial Vehicles) to spread or adopt a configuration requires to make them autonomous [1, 2]. The Collaborative Adaptative Cruise Control allows cars to evolve efficiently, regarding other cars around them [3]. The problem of network security can be approached as a Multi-Agent System problem, in order to handle the huge quantity of knowledge, but also incompleteness and inconsistency [4]. Decentralized optimization also relies on Multi-Agent concepts, in the sense that it tries to optimize an objective function determined collaboratively [5]. This topic is particularly relevant for large data [6], or when data cannot be shared between agents for privacy reasons [7].

Open Multi-Agent Systems

In our previous example of the drone formation, one scenario that could arise is the case where a “wild” drone, completely external to the system, enters the formation. In such situation, it is expected that the formation would crash. Another interesting scenario concerns the routers: one of them could fail while transmitting information, failing them to accomplish its task.

Those situations differ from the classical study of Multi-Agent Systems previously introduced in the sense that new agents entered the system, or agents left it. A system presenting this property is called *Open*. In such systems, agents may enter or leave the system at any time.

Exploring those *Open Multi-Agent Systems* is getting more and more relevant, since such systems are more and more frequent, especially in the field of decentralized methods. Indeed, as the size of the systems grows, the probability for agents to leave or join the system, to be replaced or even to experiment failures increases too. At some point, neglecting this behavior is no more suitable in the study of Multi-Agent Systems. Moreover, some systems are by nature open. Think for instance of cars eternally entering and leaving a stretch of road.

On top of it, the study of Open Multi-Agent Systems also raises economical advantages. There are expensive devices that have the particularity to fail at a low rate. Being able for systems to handle failures would allow the use of cheaper devices without that particularity, especially as, for most cases, failures are expected to ultimately happen some day or another. There also exist algorithms handling disconnections of agents in systems, but whose accuracy tend to decrease with time. Considering disconnected agents as out of the system through management of open systems in such situations can be a more efficient solution.

Work on Open Multi-Agent Systems

The study of Open Multi-Agent Systems is a rather young theory. Indeed, most of the current work, such as those presented, stand for “Closed” systems (*i.e.* systems where agents are assumed to eternally stay in the system without variations).

As discussed before, extending this work to open systems is getting more and more relevant. However, this field raises new challenges to be handled. As depicted in Fig. 1, neither the size nor the state of such systems are constant. On the first hand, each incoming agent brings in the system new information, and potentially influences the objective pursued by agents. On the other hand, each leaving agents brings out information that could be lost forever, but could also inversely leave in the system outdated information.

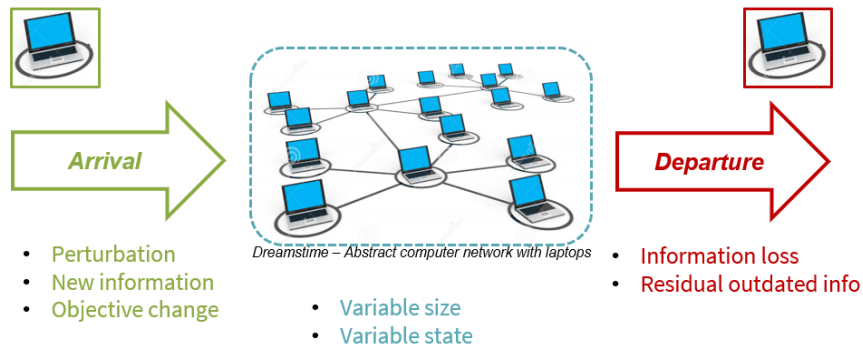


Figure 1: Representation of an Open Multi-Agent System depicting a computer network, with the implications of an agent entering or leaving the system.

This illustrates how opening Multi-Agent Systems leads to major differences, bringing no guarantee on the efficiency or validity of previous methods designed for closed systems. It has been shown in previous work that the open trait of a Multi-Agent System can imply high perturbations in the computations [8, 9]. Algorithms, and further analyses can no longer work, or dramatically decrease performances when applied on open systems. Those observations confirm that simply extending previous work viable for closed systems is not adequate for the open systems: the approach needs to be totally re-designed for them.

In the establishment of a theory around Open Multi-Agent Systems, an important part concerns the resolution and analysis of simple tasks, that can be used later in order to tackle more complex problems, and then the determination of *fundamental performance limitations* for these tasks.

Hence, the objective of this Master thesis is to consider basic problems in the framework on Open Multi-Agent Systems, and derive from them fundamental performance limitations for these problems. The interest of such limitations is to develop some kind of quality criterion on a given method to solve the basic problems they are related to. It could also allow us to develop a better understanding of the bottlenecks we could face by working on open systems.

Fundamental performance limitations for Consensus problems

One of the most basic tasks to be accomplished in many applications is the consensus problem. Such problems consist in having all agents converge to a common state defined in the system. They play a key role in most of the applications such as coordination of agents or decentralized optimization [10].

In this work, we will consider two types of consensus, defined in a system such as the one depicted in Fig. 2. We decide to work in a particular system in which agents own an intrinsic value defined by some distribution of mean μ , and of variance σ^2 . Also, the open character of the system is only represented by replacements. In other words, at each departure, there is also an arrival of agents, and then the size of the system is constant.

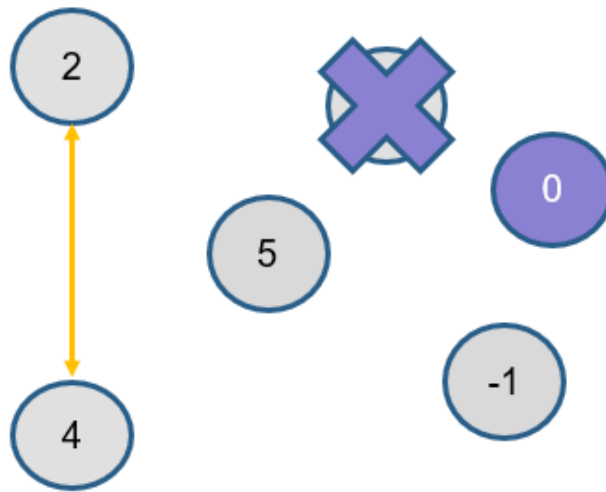


Figure 2: Small example of an Open Multi-Agent System of 5 agents, where replacements occur with a rate λ_r (in purple), and communications between pairs of agents with a rate λ_c (in yellow).

In this framework, we define two different objectives for the agents. The first one, that we call *Average Consensus Problem* consists in estimating the average of the values of the agents present in the system. The second one, that we call *Mean Consensus Problem* consists in estimating the mean μ of the distribution defining the intrinsic values of the agents.

Open Multi-Agent Systems are such that their state never stabilizes (and potentially so does the objective of the consensus). Hence, while trying to estimate this objective, an agent is expected to produce an error. Indeed, while in closed system we could reasonably expect the estimation of an agent to asymptotically converge to the right answer, the open character of open systems cannot guarantee this anymore. What we aim is then to establish the *fundamental performance limitation* for those problems, under the form of a lower bound on this error (note that for this purpose we have to define the error of estimation, which will be done later). If we manage to find a bound on the error which is always lower than the error an algorithm would provide, then this bound is a quality criterion on the efficiency of a method for solving the consensus, since it describes a minimal value the error any method cannot go under.

In order to define such bound, we have to consider an algorithm producing an estimation leading to that lower bound. In Fig. 3 are abstractly represented the sets containing all possible algorithms existing for solving a problem. In the blue region lie all the realistic algorithms, based on knowledge agents actually have access to. In the yellow region lie the algorithms relying on more information than agents really could have. Since the yellow part assumes more knowledge for the agents than the blue part, a perfectly designed algorithm in accordance with the knowledge in the yellow region produces a better estimation of the consensus than a perfect algorithm from the blue region.

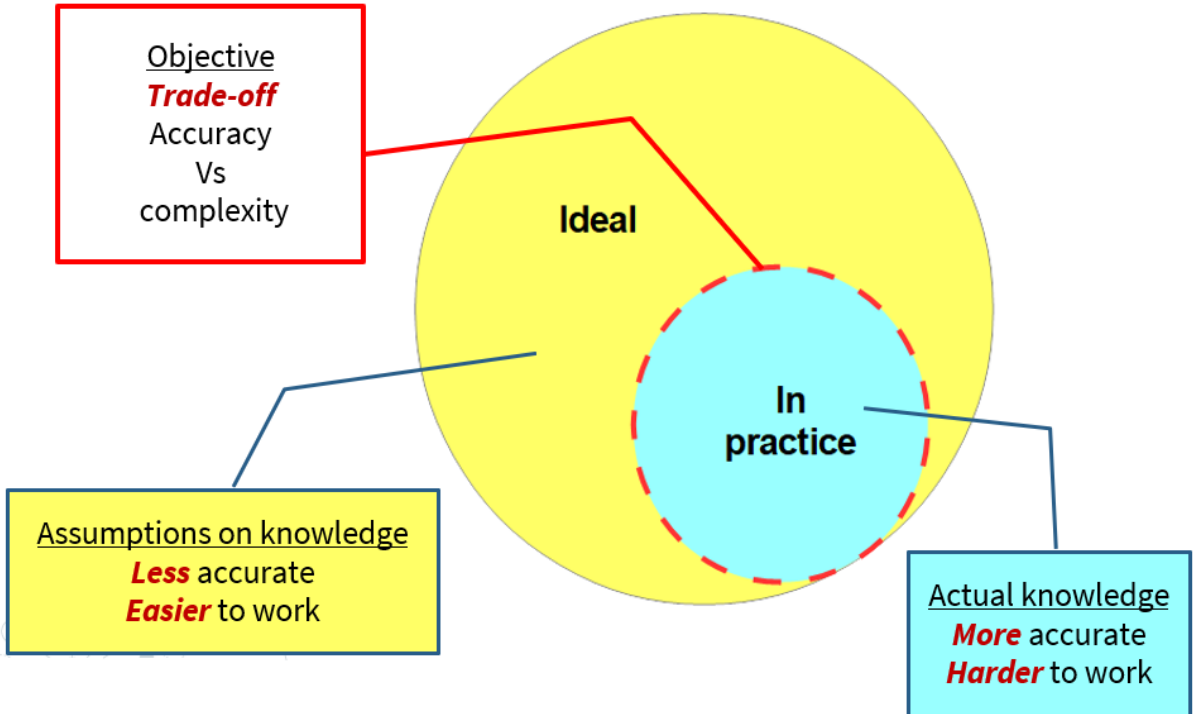


Figure 3: Abstract representation of the sets containing the algorithms to work on a problem. The blue part takes into account all the real algorithms, based on the real knowledge accessible for the agents, and can be extended to the yellow part by assuming they have some additional knowledge

Hence, the objective of this work will be to derive the error of a perfect method lying in the yellow region, but as close as possible to the blue one, in order to be as close as possible to reality, while being a lower bound. However, the closer to the blue region, the harder the computations, since less information is available. The whole challenge will thus be to find a suitable trade-off between accuracy of the lower bound, and complexity of the computations.

In Chapter 1, we will tackle the *Average Consensus Problem*. This problem has this particularity that the objective changes as agents enter or leave the system. Then, the aim of the Chapter will be to obtain a lower bound on the error of estimation of the average value of the system made by an agent for any realistic algorithm solving this problem. For this purpose, we will first present the problem, and all the assumptions that will be made to work on it. We will see that under suitable assumptions on the knowledge agents have, the error produced by an ideal algorithm is always a lower bound on the error produced by a realistic one. Then, we will compute this bound for two agents in the system. In this case, we are able to obtain the exact expression of the error for an ideal algorithm. This will be useful to present the procedure to be followed to solve such problems, and has the advantage to produce an exact and well defined bound. After that, we will extend the topic to the case of N agents. We will see that this part requires new assumptions in order to make the computations possible, and we will develop two ways of performing them to obtain the desired bound. The first one will lead to a well defined bound for any N , at the cost of a rather low precision. The second one will tend to be more general, and thus possibly more precise. We will then apply it to derive a more precise bound, but harder to compute in the sense that it relies on the resolution of a sub-problem. We will finally end this Chapter with a discussion on the assumptions defined at its beginning.

In Chapter 2, we will work with the *Mean Consensus Problem*. We will see that this problem raises different difficulties than the previous one. This time, the objective of the consensus doesn't change with the agents present in the system. This will enable us to define more precisely the error through a precise definition of the ideal algorithm that provides the agents with an estimation of their objective, and then to obtain a definition of the desired bound. We will see that this definition implies hard computations, and then, in order to get a first insight of the solution, we will perform simulations of the problem. From them, we will derive some intuition on the behavior of the error, and attempt to characterize it, in order for this to be a possible base of computations for later work.

Finally, in this document, we will explore those two problems in the framework of Open Multi-Agents Systems of constant size. Each Chapter will be related to one of those problems, and propose approaches to derive fundamental performance limitations for algorithms trying to solve them.

Chapter 1

Average Consensus Problem

1.1 Problem statement

The first problem we will work on is the Average Consensus Problem. Consider a Multi-Agent System of N agents which all own an intrinsic value x_i ($i = 1, \dots, N$), with arrivals and departures along with the process. The objective of the consensus, pursued by all agents, is to estimate the average of the intrinsic values of the agents currently present in the system. To build this estimation, agents communicate pairwise with each other as the time passes.

In order to study that problem, we place ourselves in a particular system where the intrinsic values of the agents are identically and independently distributed (i.i.d.) according to some distribution of constant mean μ and variance σ^2 . In this system, only replacements occur, according to a Poisson process of parameter λ_r , which represents the average number of replacements of an agent per unit of time. Similarly, the pairwise communications follow another Poisson process of parameter λ_c denoting the average number of communications between a given pair of agents per a unit of time. Such a particular system involved in this consensus problem is depicted in Fig. 1.1.

Due to the open character of the system, agents produce an error as they estimate the average. In this Chapter, we will try to establish a lower bound on this error, in a situation where we consider the system has already been running for a long time, so that it has reached a steady state. For this purpose, it is required to define that error, through some quality criterion for the estimation made by an agent. We arbitrarily choose to define it as the Mean Square Error on the true average.

1.1.1 Assumptions on knowledge

The more information the agents have access to, the more accurate their estimation, and thus the lower the error induced. In order to obtain a lower bound on the error for any algorithm trying to solve this consensus problem, we want to obtain the error induced by some ideal algorithm relying on more information than a realistic one could. Such an algorithm should provide some kind of best possible estimation of the average, given the knowledge agents possess.

To do so, we provide all agents in the system with more knowledge than they would have in a real-life situation. If we manage to find a lower bound on the error produced by an ideal algorithm relying on that knowledge, then this is necessarily also a lower bound on any less accurate realistic algorithm. This is illustrated in Fig. 3, where this ideal algorithm lies in the region of the improved algorithms (in yellow), which, provided it is effectively ideal with its information, is by definition better than a realistic algorithm (from the blue region).

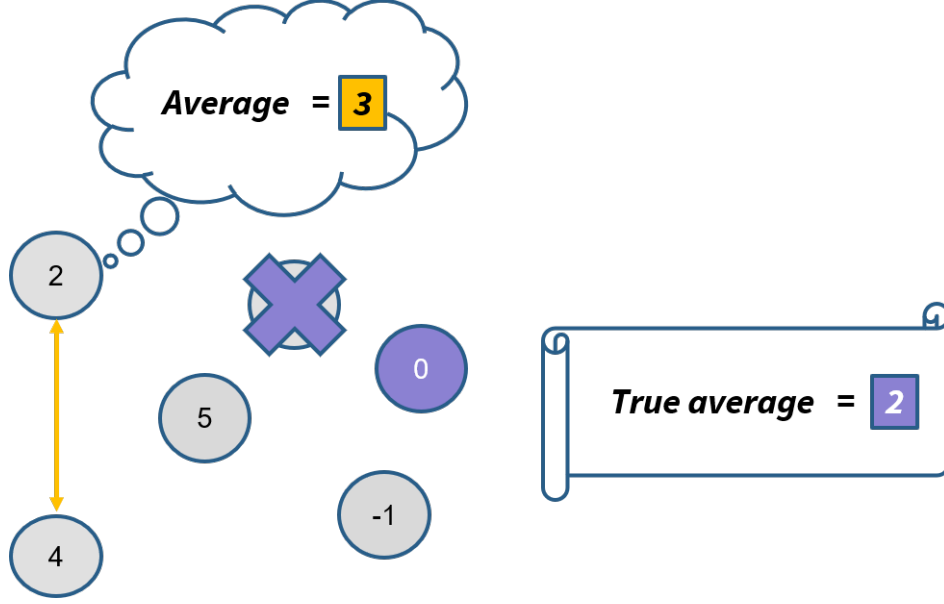


Figure 1.1: Small example of an Open Multi-Agent System of 5 agents, where replacements occur with a rate λ_r (in purple), and communications between pairs of agents with a rate λ_c (in yellow), in an Average Consensus Problem

Hence, we will consider for this Chapter, that the agents in the system know the its size N , and the fact that it is constant. Moreover, agents also know the replacement and communication processes, and then the parameters defining them: λ_r and λ_c . Finally, agents have some oracle giving them the constant parameters defining the distribution of their intrinsic values: the mean μ and the variance σ^2 . For the sake of simplicity, and without loss of generality, we will consider for this Chapter that $\mu = 0$. Indeed, the results that will follow can easily be adapted for $\mu \neq 0$.

1.1.2 Definition of the estimator and the error

With their improved knowledge, we expect agents to build the best possible estimation of the average, so that the error we will derive will be a lower bound of a realistic one. Considering the definition of the error as the Mean Square Error, it means we assume agents know some ideal algorithm that produces, given what they know of the system, the estimation minimizing the Mean Square Error. Thus, the estimation y_i made by agent i is actually the *Minimum square error estimator* of the average $\bar{x}$.

$$\begin{aligned}
 y_i &= \arg \min \left(\mathbb{E} \left[(\bar{x} - y_i)^2 \mid \text{Knowledge of agent } i \right] \right) \\
 &\Leftrightarrow 0 = \frac{d}{dy_i} \left(\mathbb{E} \left[(\bar{x} - y_i)^2 \mid \text{Knowledge of agent } i \right] \right) \\
 &\Leftrightarrow 0 = 2y_i - 2\mathbb{E} [\bar{x} \mid \text{Knowledge of agent } i] \\
 &\Leftrightarrow y_i = \mathbb{E} [\bar{x} \mid \text{Knowledge of agent } i]
 \end{aligned}$$

Then, the error we would like to bound is given by the Mean Square Error definition

$$\mathbb{E} [\bar{x} - \mathbb{E} [\bar{x} \mid \text{Knowledge of agent } i]]^2 \quad (1.1)$$

1.2 Bound for two agents in the system

Let us start with a quite simple case, in order to get used with the problem, and with the way of tackling it. We consider a system as the one presented above, constituted of two agents. As presented in the problem statement, agents are subject to replacements, and communications. In this particular context, the only communications that can happen are between agents 1 and 2.

Our objective here is to compute the error produced by any of those agents in its estimation of the average of the system, given by

$$\bar{x} = \frac{x_1 + x_2}{2} \quad (1.2)$$

Since both agents are interchangeable, we can place the focus on agent 1 without loss of generality, and compute the error it makes on the estimation of $\bar{x}$, as the expected error made by any agent:

$$\mathbb{E} [\bar{x} - \mathbb{E} [\bar{x} \mid \text{Knowledge of agent 1}]]^2 \quad (1.3)$$

In this specific situation of having two agents present in the system, it is easy to list all the possible states of the system at a given time, from the point of view of agent 1. Indeed, either it doesn't know agent 2, and estimates it the best it can given its general knowledge, either it knows it from the last communication they had, and estimates it based on its knowledge about a potential replacement since they last communicated.

Since we are able to list all those cases, it is possible to build the exact estimation of the average, and thus the exact expected error agent 1 makes under those assumptions of knowledge. In the following parts, we are going to compute this error and analyze it. Since we work with an ideal algorithm under big assumptions on the knowledge of the agents, this error will be a lower bound on the error we would obtain with any more realistic algorithm, relying on less information.

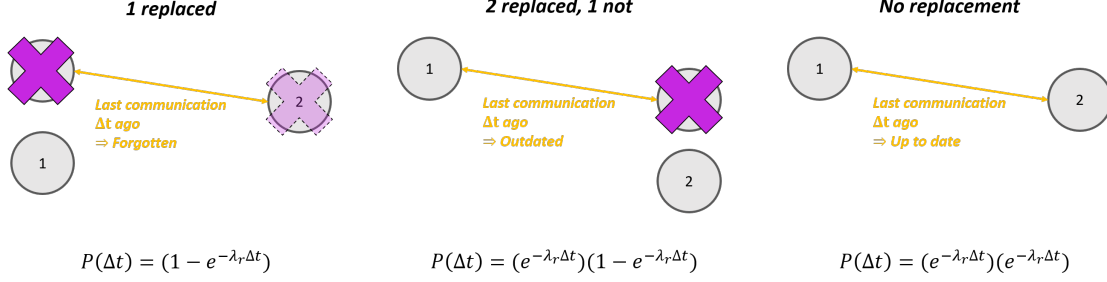
1.2.1 Computation

We want to obtain the expected error induced by the estimation of the average of the system. As a reminder, we consider the error obtained when the system is at a steady state, defined as the Mean Square Error on the average $\bar{x}$. To obtain the error at a steady state, we will need to consider every possible configurations of the system, and then average the error each configuration implies, weighted by its probability to happen.

We mentioned before that with two agents, it is quite easy to list all possible states of the system, given the elapsed time since the last communication between the agents.

Let the time since the last communication between the agents be Δt . This was the last time agent 1 updated its knowledge about agent 2. Since then, during Δt , it has either been replaced at least once, and has then no more knowledge about agent 2 no matter what happened to it, given the fact that the last communication happened before that replacement (denote that case R_1 , depicted in Fig. 1.2a) ; either it has not been replaced (N_1). In this second case, there are two more cases to be considered: agent 2 may have been replaced at least once, leaving the information about it outdated (denote this case R_2 , depicted in Fig. 1.2b), or not (N_2 , Fig. 1.2c).

Then, we can build the estimates of the average made by agent 1, and then the corresponding error, for each of those cases. We obtain from them the expected error given the time of the last



(a) R_1 : Agent 1 replaced, no matter agent 2 (b) $N_1 \cap R_2$: Agent 1 not replaced, agent 2 replaced (c) $N_1 \cap N_2$: Agent 1 not replaced, nor agent 2

Figure 1.2: All possible system configurations in a time lapse Δt since the last communication, from the point of view of agent 1.

communication Δt , by summing up those errors weighted by the probability the corresponding case happens.

We find those probabilities thanks to the mechanism defining the replacements: a Poisson process of parameter λ_r . In such processes, the probability for an event to happen is given by an exponential law. Hence, for both agents (since they are interchangeable), the probability to be replaced in a time lapse Δt is given by $p_R(\Delta t) = 1 - e^{-\lambda_r \Delta t}$, and the probability to not be replaced by $p_N(\Delta t) = e^{-\lambda_r \Delta t}$.

We can finally build the estimation made by agent 1 of the average $\bar{x}$. The last thing to be settled is its estimation of the value of the second agent. Denote by X the random variable defining the intrinsic value of agent 2 when agent 1 doesn't have any information about it. By definition, X follows the same distribution as x_1 and x_2 . Hence, we have $\mathbb{E}X = 0$, since we set the mean $\mu = 0$, and $\mathbb{E}[X^2] = \mu^2 + \sigma^2 = \sigma^2$.

Then, we have

$$\mathbb{E}[x_2(t)|x_2(t - \Delta t)] = p_R(\Delta t)\mathbb{E}[X] + p_N(\Delta t)x_2(t - \Delta t) = e^{-\lambda_r \Delta t}x_2(t - \Delta t) \quad (1.4)$$

and then

- $\mathbb{E}[\bar{x}(t)|R_1] = \mathbb{E}\left[\frac{x_1(t)+X}{2}\right] = \frac{x_1(t)}{2}$
- $\mathbb{E}[\bar{x}(t)|N_1] = \mathbb{E}\left[\frac{x_1(t)+x_2(t)}{2}|x_2(t - \Delta t)\right] = \frac{x_1(t)}{2} + \frac{\mathbb{E}[x_2(t)|x_2(t - \Delta t)]}{2} = \frac{x_1(t)+x_2(t - \Delta t)e^{-\lambda_r \Delta t}}{2}$

From those, we can derive the error induced in each of the three cases listed previously. In order to lighten the notations, we will use as notation for the probability that no replacement happened $p_N := e^{-\lambda_r \Delta t}$, and also use $x_i := x_i(t - \Delta t)$ (with $i = 1, 2$). Note also that, due to the uncorrelation of all intrinsic values, we have $\mathbb{E}[x_i x_j] = \mathbb{E}[x_i]\mathbb{E}[x_j]$ for $i \neq j$.

- $\mathbb{E}[(\bar{x} - y_1)^2|R_1] = \mathbb{E}\left[\frac{x_1}{2} - \frac{x_1+X}{2}\right]^2 = \frac{\mathbb{E}[X^2]}{4} = \frac{\sigma^2}{4}$
- $\mathbb{E}[(\bar{x} - y_1)^2|N_1 \cap R_2] = \mathbb{E}\left[\frac{x_1+p_N x_2}{2} - \frac{x_1+X}{2}\right]^2 = \frac{p_N^2 \mathbb{E}[x_2^2] - 2p_N \mathbb{E}[x_2 X] + \mathbb{E}[X^2]}{4} = \frac{(p_N^2 + 1)}{4} \sigma^2$
- $\mathbb{E}[(\bar{x} - y_1)^2|N_1 \cap N_2] = \mathbb{E}\left[\frac{x_1+p_N x_2}{2} - \frac{x_1+x_2}{2}\right]^2 = \frac{(p_N - 1)^2 \mathbb{E}[x_2^2]}{4} = \frac{(p_N - 1)^2}{4} \sigma^2$

The next step is thus to sum up all weighted cases as follows.

$$\begin{aligned}\mathbb{E}[(\bar{x} - y_1)^2 | \Delta t] &= p_R(\Delta t) \mathbb{E}[(\bar{x} - y_1)^2 | R1] \\ &\quad + p_N(\Delta t) \left(p_R(\Delta t) \mathbb{E}[(\bar{x} - y_1)^2 | N1 \cap R2] + p_N(\Delta t) \mathbb{E}[(\bar{x} - y_1)^2 | N1 \cap N2] \right) \\ &= (1 - p_N^3) \frac{\sigma^2}{4}\end{aligned}$$

Now, to obtain the steady state, we need to integrate this result weighted by the probability that the last communication occurred Δt ago. Just like for the replacements, the communications between any pair of agents is governed by an exponential law of parameter λ_c . The probability density function defining the probability that a communication occurred between agents 1 and 2 is then given by the exponential law as $f_C(t) = \lambda_c e^{-\lambda_c t}$.

By integrating, we obtain

$$\begin{aligned}\mathbb{E}[\bar{x} - y_1]^2 &= \int_0^\infty \lambda_c e^{-\lambda_c t} \left(1 - e^{-3\lambda_r t} \right) \frac{\sigma^2}{4} dt \\ &= \frac{\lambda_c \sigma^2}{4} \left[\frac{e^{-\lambda_c t}}{-\lambda_c} - \frac{e^{-(\lambda_c + 3\lambda_r)t}}{-(\lambda_c + 3\lambda_r)} \right]_0^\infty = \frac{\lambda_c \sigma^2}{4} \left(\frac{1}{\lambda_c} - \frac{1}{\lambda_c + 3\lambda_r} \right)\end{aligned}$$

A few algebraic manipulations finally give us the final form of the bound.

$$\mathbb{E}[\bar{x} - y_1]^2 = \frac{\sigma^2}{4} \left(1 - \frac{3\lambda_r}{\lambda_c + 3\lambda_r} \right) = \frac{\sigma^2}{4} \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r}} \right) \quad (1.5)$$

Hence, we can conclude that the error obtained in a system of two agents is always bigger than this value, as presented in Theorem 1.

Theorem 1. *In a system constituted of two agents trying to solve an Average Consensus Problem, with a distribution of null mean and of variance σ^2 for the intrinsic values of the agents, and where replacements occur at a rate λ_r and communications at a rate λ_c , then the Mean Square Error produced by the estimation y_i made by an agent i is bounded by below, and*

$$\mathbb{E}[\bar{x} - y_i]^2 \geq \frac{\sigma^2}{4} \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r}} \right)$$

1.2.2 Observations

We finally derived the Mean Square Error made by agent 1 when estimating the average of the system, according to the ratio of the rates λ_c/λ_r . This error is depicted in Fig. 1.3. As a reminder, the assumptions made on the accessible knowledge of the agents to build their estimation of the average used to derive this bound are such that they have more information than a real-life algorithm would allow. Hence, we currently lie in the yellow part of Fig. 3, and this error is supposed to be a lower bound on the error obtained by using any realistic method (in the blue part).

If we proceed to a few sanity checks in order to validate the asymptotic behavior, we can conclude that this bound behaves as it is supposed to. Indeed, when $\lambda_c/\lambda_r \rightarrow \infty$, or in other words when there are way more communications between the agents than there are replacements, we expect, asymptotically, the error to go to 0, since this case comes to close more and more

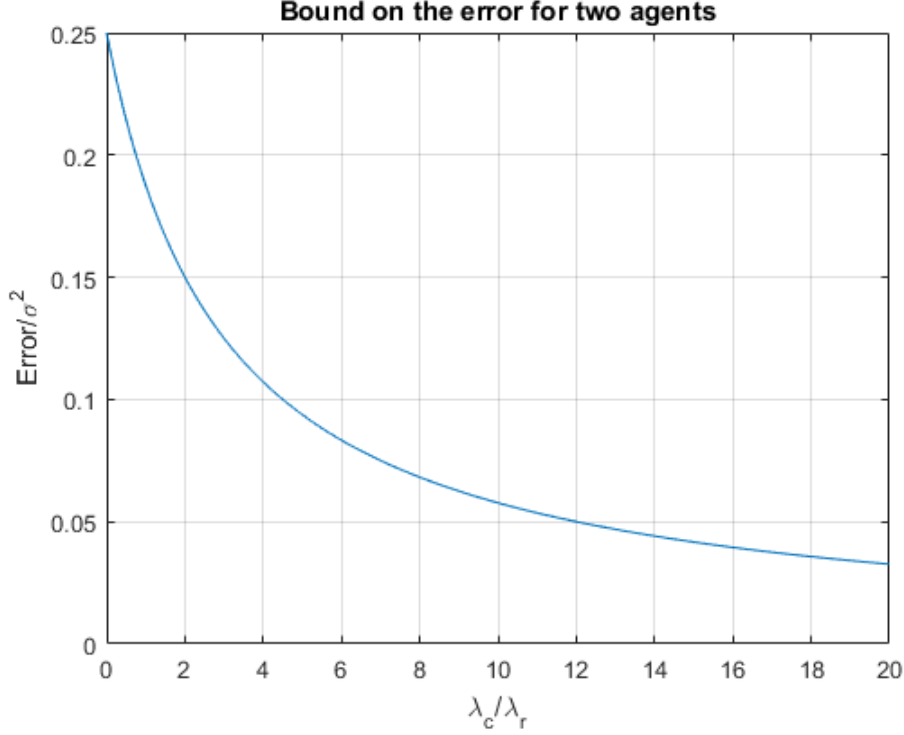


Figure 1.3: Bound on the error obtained for two agents in the system

the system the closer to ∞ . On the opposite, when $\lambda_c/\lambda_r \rightarrow 0$, or in other words when the replacements completely overtake the communications, then we expect to have an error similar to a case when we never know the value of the other agent in the system, and it gives in any case an error $\mathbb{E} \left[\frac{x_1}{2} - \frac{x_1+X}{2} \right]^2 = \mathbb{E} \left[\frac{X}{2} \right]^2 = \sigma^2/4$, which is the value of the bound when $\lambda_c/\lambda_r = 0$.

At the end of the day, this first Section allowed us to compute an exact error for a simple case, and to get familiar with the procedure to compute such bounds. We will see in the next Sections that the other cases raise new difficulties, that will require to establish new assumptions.

1.3 Ping model

When considering the Average Consensus Problem with N agents in the system, the problem gets more sophisticate. Unlike the case with two agents, listing all the possible states of the system at a given time requires to take into account a lot of factors. Indeed, while enumerating those states was manageable by knowing the last communication time with two agents, it requires a better understanding of the information spreading mechanism in the system for N agents. Knowing the last time an agent was involved in a communication doesn't guarantee anymore a knowledge update about all the other agents: only information known by the agents concerned by the communication is available for a potential update of knowledge.

Hence, building the best estimation obtained with an ideal algorithm, given the knowledge of the agents, reveals to be hard. In order to make those computations possible, we choose to relax the complexity of the information spreading mechanism, by giving agents even more information about the system state than they actually have at each communication. Of course, this way of proceeding is not the only possibility: in the next Section, we will relax this difficulty differently.

Under those additional assumptions, we will build the ideal algorithm providing the best possible estimation of the average, in the sense of the Mean Square Error induced by this estimation. Since it relies on more information than the real problem we attempt to solve normally provides, as depicted in Fig. 3, we expect to obtain a better estimation of the average, and then a lower error than what a normal algorithm would provide. Furthermore, since the error we will obtain is a lower bound on the error of the standard problem under the assumptions of knowledge described in Section 1.1.1, which is already supposed to lead to a lower bound on the error of any algorithm, this should also be a lower bound for any algorithm.

Then the assumption we are going to make for this Section is rather strong. We will assume that, at each communication, both involved agents update their knowledge about the whole system. Under this assumption, agents clearly have access to more information they should in the not-relaxed problem, and thus by building the best possible estimation through the ideal algorithm in this context, they are expected to compute a better estimation.

We call this method the *Ping model*, as a reference to the *ping* software, which tests the accessibility of a machine in a network, and more specifically to its *fping* improvement which tests it for several machines at the same time.

In this Section, we will first provide of procedure of computation of the bound on the error, by defining the estimation agents make based on their knowledge in order to minimize the error. Then, we will compute it in the general case of having N agents in the system, and study the obtained result. Considering the strong assumption we just added, we expect the obtained bound to be really lower than reality, but as we will see it has the advantage to be rather easy to compute and to provide an explicit form of the bound.

1.3.1 Computation procedure

First, we will describe the procedure that we will follow in order to compute the bound on the error under those assumptions. As for the 2 agents case, we attempt to derive a steady state of the error, by integrating over all the possible configurations of the system the error those configurations induce.

Defining the estimate

The first thing to be done is to establish the estimate of the average made by an agent under the assumptions. Indeed, given its knowledge about the system, an agent is expected to build the estimation of the average that minimizes the Mean Square Error.

In our situation, agents have access to the general knowledge of the system (such as the communication and replacement rates λ_c and λ_r , or information related to the distribution), and also have knowledge about the whole state of the system at the time they last communicated. From this, the first step in the computation is to obtain the estimation made by a given agent i , provided by an ideal algorithm given this knowledge

$$y_i = \mathbb{E} [\bar{x} \mid \text{Knowledge of agent } i] \quad (1.6)$$

Listing all cases for a given time

The second thing to be done, based on the estimation of the average given the knowledge of the agent, is to list the cases that can happen in this knowledge configuration, and derive the

corresponding errors, in order to build the expected error in this configuration.

In this model, the assumption on the information spreading mechanism defines those configurations. Indeed, agents build their estimate based on general knowledge, which is the same at any time, and based on their current knowledge about the system state, defined by the last communication they were involved in.

To illustrate this, consider a given agent, and that the system is currently at time t_0 . The last communication involving our agent happened at time t_c , Δt ago, as depicted in Fig. 1.4. This elapsed time defines a configuration of the system under the Ping assumption, as we just said. During that time, several cases could happen as agents can still have been replaced. Such a replacement is depicted by the time $t_{r,i}$ on the figure for instance. However, no other relevant communication can happen in that time.

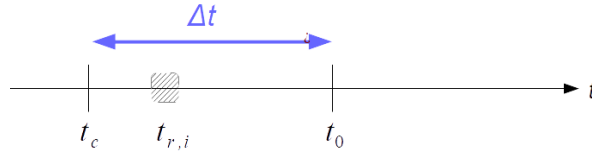


Figure 1.4: Possible succession of events in a time lapse Δt starting at the last communication of an agent.

Let Ω be the set containing all the possible series of events that could take place in the elapsed time Δt since the last communication. In other words, a configuration $\omega \in \Omega$ is a possible succession of replacements that could take place during Δt . Then we can define the error in the case where the last communication happened Δt ago as follows

$$\mathbb{E}[(\bar{x} - y_i)^2 | \Delta t] = \sum_{\omega \in \Omega} P_{\omega}(\Delta t) \mathbb{E}[(\bar{x} - y_i)^2 | \Delta t, \omega] \quad (1.7)$$

where $P_{\omega}(\Delta t)$ is the probability that the configuration $\omega \in \Omega$ happened in the time lapse Δt .

Getting the steady state

From the error according to the time since the last communication, we can obtain the steady state of the error by considering all possible values of that time. Hence, defining some probability density function $f_C(\Delta t)$ associated with the probability of having the last communication some time Δt ago, we can obtain the final expression for the bound on the error

$$\mathbb{E}[\bar{x} - y_i]^2 = \int_0^{\infty} f_C(\Delta t) \mathbb{E}[(\bar{x} - y_i)^2 | \Delta t] d\Delta t \quad (1.8)$$

This concludes the procedure that will be followed in order to obtain our first bound on the error for the Average Consensus Problem.

1.3.2 Results

Let us finally compute the error with the procedure we just defined for the case of having N agents in the system under the assumptions on the knowledge of the agents, and those from the Ping model. To do so, we will consider each step proposed in the procedure, and apply it.

The same way we did for the case of two agents, we will place the focus on agent 1 without loss of generality, provided that all agents are interchangeable.

Defining the estimate

First, it is required to define the estimation of the average of the system given by the ideal algorithm, based on the knowledge of agent 1 about the current state of the system. There are two cases defining the knowledge of the agent: either it has already communicated at least once since it is in the system, or either it has not. Another way to formulate it is to say that since the last communication involving agent 1, it may have been replaced at least once (denote that case R_1), and has no more knowledge about the other agents than its general information about the system, or it may have not been replaced (denote it N_1), and thus builds its estimation based on the information it gathered in the last communication.

In the second case, agent 1 considers the possibility of a replacement of the other agents during the time since the last communication Δt . Hence, we can write how agent 1 estimates the intrinsic value of any other agent j based on its knowledge from Δt ago, with $p_N(\Delta t) = e^{-\lambda_r \Delta t}$ the probability that any agent has not been replaced in a time Δt given by the Poisson process, and $p_R(\Delta t) = 1 - e^{-\lambda_r \Delta t}$ the probability it has at least once.

It provides, by denoting with the random variable X the unknown intrinsic value of any agent, subject to the same distribution and assumptions:

$$\mathbb{E}[x_j(t)|x_j(t - \Delta t)] = p_N(\Delta t)\mathbb{E}[X] + p_R(\Delta t)x_j(t - \Delta t) = e^{-\lambda_r \Delta t}x_j(t - \Delta t) \quad (1.9)$$

Hence, by denoting once more $p_N := e^{-\lambda_r \Delta t}$ to lighten the notations, we obtain

$$\begin{aligned} \bullet \mathbb{E}[\bar{x}|R_1] &= \mathbb{E}\left[\frac{x_1(t) + \sum_{j=2}^N X}{N}\right] = \frac{x_1(t)}{N} + \sum_{j=2}^N \frac{\mathbb{E}X}{N} = \frac{x_1(t)}{N} \\ \bullet \mathbb{E}[\bar{x}|N_1] &= \mathbb{E}\left[\frac{\sum_{j=1}^N x_j(t)}{N}\right] = \frac{x_1(t)}{N} + \sum_{j=2}^N \frac{\mathbb{E}[x_j(t)|x_j(t - \Delta t)]}{N} = \frac{x_1 + \sum_{j=2}^N p_N x_j(t - \Delta t)}{N} \end{aligned}$$

Listing all cases for a given time

Listing all the possible cases to build the expected error given the time of the last communication is a problem that gets harder as N grows. In order to solve it, let us start with some little values of N , so that we can generalize it to any N .

When $N = 2$, we actually come back to the exact same case as the one with two agents we developed in the previous Section. Indeed, since the last communication between the agents, only three cases arise: agent 1 could be replaced at least once and loose all its knowledge, or not. If not, then agent 2 could have been replaced at least once, or not, leading to different errors. Those are the same three cases as presented in Section 1.2.1 when computing the bound for two agents, and thus lead to the same expected error $\mathbb{E}[(\bar{x} - y_1)^2|\Delta t] = \frac{1}{4}(1 - p_N^3)\sigma^2$. Similarly, the case with three agents has also been developed, and leads to an error $\mathbb{E}[(\bar{x} - y_1)^2|\Delta t] = \frac{2}{9}(1 - p_N^3)\sigma^2$ (the computations being quite heavy, details are left to the reader in Appendix A). It is tempting, considering it, to assume we can extend the expression to the N agents case as follows

$$\mathbb{E}[(\bar{x} - y_1)^2|\Delta t] = \frac{N-1}{N^2} (1 - e^{-3\lambda_r \Delta t}) \sigma^2. \quad (1.10)$$

It is proven in Appendix B that the above equation is valid for the Ping model with N agents. The proof relies on the fact that intrinsic values are i.i.d., and thus that we can separate the contribution of each agent of the system in the estimation of the error. This idea will be more detailed in a further Section.

Getting the steady state

From this expression, all that is left to be done is to integrate over the possible values of Δt . To do so, let us build the probability of having a communication in the time lapse Δt , based on the Poisson process defining pairwise communications.

Consider, from the exponential law defining the pairwise communications, the probability that no communication occurs in a time t between agent 1 and any other agent: $e^{-\lambda_c t}$. This probability holds for only one potential partner. We can extend it to the probability that there hasn't been any communication with any other agent. Since there are $N - 1$ potential partners for a communication, and since communications are independent processes, this probability is given by $(e^{-\lambda_c t})^{N-1} = e^{-(N-1)\lambda_c t}$. Hence, the probability that there has been at least one communication with any potential partner is complementary to the previous one, and is given by $1 - e^{-(N-1)\lambda_c t}$. We can then derive the related probability density function.

$$f_C(t) = (N - 1)e^{-(N-1)\lambda_c t} \quad (1.11)$$

Integrating, it gives

$$\begin{aligned} \mathbb{E} [\bar{x} - y_1]^2 &= \int_0^\infty (N - 1)\lambda_c e^{-(N-1)\lambda_c t} (1 - e^{-3\lambda_r t}) \frac{N - 1}{N^2} \sigma^2 dt \\ &= \frac{(N - 1)^2}{N^2} \lambda_c \sigma^2 \left[\frac{e^{-(N-1)\lambda_c t}}{-(N - 1)\lambda_c} - \frac{e^{-[(N-1)\lambda_c + 3\lambda_r]t}}{-[(N - 1)\lambda_c + 3\lambda_r]} \right]_0^\infty \\ &= \frac{N - 1}{N^2} \sigma^2 \left(1 - \frac{(N - 1)\lambda_c}{(N - 1)\lambda_c + 3\lambda_r} \right) \end{aligned}$$

A few algebraic manipulations lead to the final form of our definitive bound for the Ping model with N agents.

$$\mathbb{E} [\bar{x} - y_1]^2 = \frac{N - 1}{N^2} \left(\frac{1}{1 + \frac{N-1}{3} \frac{\lambda_c}{\lambda_r}} \right) \sigma^2 \quad (1.12)$$

Equation (1.12) is the final form of the bound on the error of estimation of the average value of the system made by an agent, under the Ping model assumptions. This expression fits correctly with the case with two agents, and as predicted has the major advantage to be rather easy to obtain, and given explicitly. We will see in further investigations that we can't always expect to have a proper expression for the bound. We have then showed that this expression of the error always bounds by below the error we would have obtained using some realistic algorithm, as presented in Theorem 2.

Theorem 2. *Given a system constituted of N agents trying to solve an Average Consensus Problem, with a distribution of null mean and of variance σ^2 defining the intrinsic values of the agents, and where replacements occur at a rate λ_r and pairwise communications at a rate λ_c , then the Mean Square Error produced by the estimation y_i made by an agent i is bounded by below, and*

$$\mathbb{E} [\bar{x} - y_i]^2 \geq \frac{N - 1}{N^2} \left(\frac{1}{1 + \frac{N-1}{3} \frac{\lambda_c}{\lambda_r}} \right) \sigma^2$$

In the next Section, we will discuss the quality and behavior of this bound.

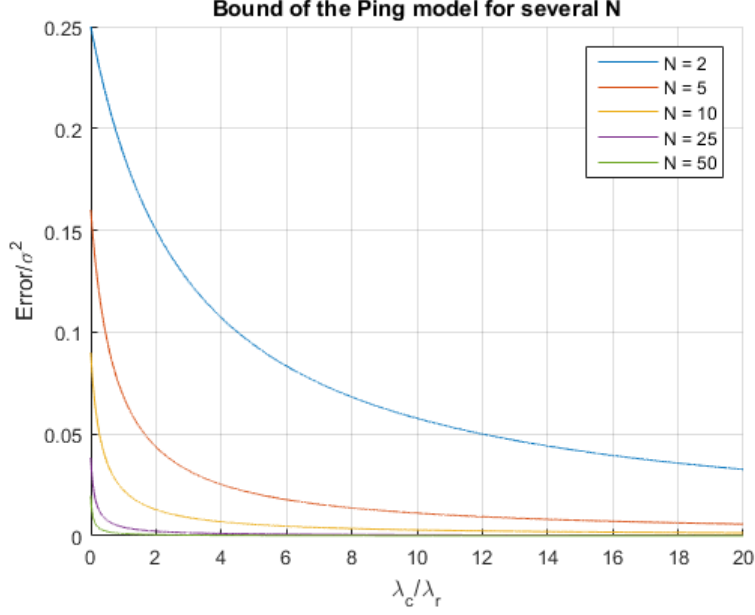


Figure 1.5: Evolution of the bound on the error with the ratio of the rates, for $N = 2, 5, 10, 25, 50$. The bound is normalized by the variance, so that the curve corresponds here to a unit variance. One sees that the bound decrease when the communications overtake the replacements, and that the bound is lower the more agents in the system.

1.3.3 Discussion

We show in Fig. 1.5 the bound we just obtained for the Ping model for several numbers of agents in the system, going from two to fifty. This curve is supposed to bound by below the expected error made by any algorithm to estimate the average intrinsic value of the system.

Sanity checks

We will quickly go over some sanity checks, in order to evaluate the consistency of the bound. Those checks do not attempt to prove the validity of the method, but are a necessary (but not sufficient) condition for the bound to be acceptable.

- $\lambda_c \gg \lambda_r \Rightarrow \text{Error} \rightarrow 0$: when the communication rate significantly overtakes the replacement rate (or similarly $\lambda_c \rightarrow \infty$), we expect the error to go to 0, since agents are assumed to permanently updating their knowledge of the system, as if the system was getting close.
- $\lambda_r \rightarrow 0$: in a similar way, when there is no replacement at all, we expect the error to go to 0, since the system never changes: we actually come back to a close system.
- $\lambda_c \ll \lambda_r \Rightarrow \text{Error} \rightarrow \frac{N-1}{N^2}\sigma^2$: inversely, when the replacements overtake the communications (or similarly $\lambda_c \rightarrow 0$ or $\lambda_r \rightarrow \infty$), then the best estimation made by an agent is the mean, and the error becomes $\mathbb{E}[\bar{x} - y_i]^2 = \mathbb{E}\left[\sum_{j=2}^N X_j/N\right]^2 = \frac{N-1}{N^2}\sigma^2$ (since all agents are uncorrelated). This corresponds to the value of the bound for $\lambda_c/\lambda_r = 0$.
- $N \text{ increases} \Rightarrow \text{Error decreases}$: when the number of agents in the system increases, the probability of a communication, and then of an update of knowledge, to happen increases. Thus, so does the quality of the estimation, and the error is expected to decrease.

Some remarks

The sanity checks being satisfied, we can be rather confident for the validity of the bound. However, as we mentioned in the beginning of this Section, the assumptions made on the information spreading mechanism are really strong. The lower bound we obtained here has the advantage to be easy to compute and to use, but shows itself imprecise, especially when the number of agents increases. Fig. 1.5 proves that quickly, as N grows, the bound tends to 0, no matter the rates, which makes the bound useless in those circumstances.

Still, this is a relevant first approximation. And in what follows, we will try to generalize the methodology developed here, in order to build more precise bounds.

1.4 Development of a general method

In the previous Section, we had to deal with the problem of having N agents in the system. We realized that, in this context, it is hard to proceed the same way as we did for the case with two agents, since the knowledge learned by an agent through a communication is hard to estimate. To deal with this difficulty, we added a strong assumption on the communication mechanism, making the computations easier. That way we managed to obtain a bound quite easily, but also quite inaccurate.

In this Section, we intend to develop a method to estimate a bound without that strong assumption. Hence, we consider here that the information that has an agent concerning the other ones in the system, on which the ideal algorithm bases itself, is only obtained through communications with agents that have it. More precisely, when agents communicate, they exchange their knowledge so that they keep the most up to date of what they have together. Then, they build the best possible estimation with the ideal algorithm based on this knowledge.

As the problem of defining this knowledge about the other agents remains, we will have to make new assumptions somewhere else. To do so, we will investigate more in depth the definition of the estimation of the average, and derive a new procedure to evaluate the error. In the process, as the computations remain hard, we will find a new opportunity to add a simplifying assumption, that will lead to a lower bound on the error. The ultimate goal will then, through this new method, to improve the quality of the bound we obtained with the previous method.

1.4.1 New procedure

We start by investigating the definition of the estimator of the average, used by some agent i . From this, we will build a new procedure to estimate a bound on the error.

Individual contribution of the agents

We defined in Section 1.1.2 the estimator of the average used by any agent. In order to detail what will follow, let us develop this definition further.

The estimation made by agent i is given by

$$y_i(t) := \mathbb{E}[\bar{x}(t) \mid \text{Knowledge of } i] = \mathbb{E}\left[\frac{\sum_{j=1}^N x_j(t)}{N} \mid \text{Knowledge of } i\right] = \frac{1}{N} \sum_{j=1}^N \hat{x}_j(t) \quad (1.13)$$

where we denote by $\hat{x}_j(t)$ the estimation of the value of agent j made by agent i , based on its knowledge about it at time t . One can develop this estimation, based on the fact that

replacements are governed by an exponential law of parameter λ_r (from the Poisson process). Hence we can write

$$\hat{x}_j(t) = \mathbb{E}[x_j(t)|x_j(t-T_j)] = (1 - e^{-\lambda_r T_j})\mu + (e^{-\lambda_r T_j})x_j(t-T_j) = e^{-\lambda_r T_j}x_j(t-T_j) \quad (1.14)$$

where T_j denotes the elapsed time since the last update of knowledge about agent j .

We can then develop the definition on the Mean Square Error which we use to estimate the error induced by the above estimation of the average, defined in Section 1.1.2. For the sake of simplicity, we will lighten the notations by defining $x_j := x_j(t)$ for any $j = 1 \dots N$.

$$\begin{aligned} \mathbb{E}[\bar{x} - y_i]^2 &= \mathbb{E}\left[\frac{\sum_{j=1}^N x_j}{N} - \frac{\sum_{j=1}^N \hat{x}_j}{N}\right]^2 = \frac{1}{N^2} \mathbb{E}\left[\sum_{j=1}^N (x_j - \hat{x}_j)\right]^2 \\ &= \frac{1}{N^2} \mathbb{E}\left[\sum_{j=1}^N (x_j - \hat{x}_j)^2 + \sum_{j \neq k} (x_j - \hat{x}_j)(x_k - \hat{x}_k)\right] \\ &\quad \left| \text{From uncorrelation of } x_j \text{ and } x_k \text{ (since independent): } \mathbb{E}[x_j x_k] = \mathbb{E}[x_j] \mathbb{E}[x_k] \right. \\ &= \frac{1}{N^2} \mathbb{E}\left[\sum_{j=1}^N (x_j - \hat{x}_j)^2\right] + \frac{1}{N^2} \sum_{j \neq k} \mathbb{E}[x_j - \hat{x}_j] \mathbb{E}[x_k - \hat{x}_k] \\ &\quad \left| \text{Since } \mathbb{E}[x_j - \hat{x}_j] = \mathbb{E}[x_j] - e^{-\lambda_r T_j} \mathbb{E}[x_j] = 0 \right. \\ &= \frac{1}{N^2} \sum_{j=1}^N \mathbb{E}[(x_j - \hat{x}_j)^2] \end{aligned}$$

Hence, the error we are trying to bound is actually the sum of the individual errors made on each agent's estimation, as presented in equation (1.15). This will allow us to make some major assumptions in the definition of the steady state of the error.

$$\mathbb{E}[\bar{x}(t) - y_i(t)]^2 = \frac{1}{N^2} \sum_{j=1}^N \mathbb{E}[(x_j(t) - e^{-\lambda_r T_j} x_j(t-T_j))^2] \quad (1.15)$$

Steady state of the error

One can define the steady state of the error of estimation as in equation (1.16)

$$\mathbb{E}[\bar{x} - y_i]^2 = \int_{\omega \in \Omega} f_\omega \mathbb{E}[(\bar{x} - y_i)^2 | \omega] \quad (1.16)$$

where Ω stands for the set of all possible configurations of the system, and f_ω the probability density function related to the probability for the system to lie in a configuration $\omega \in \Omega$.

Consider the system at some time t . One configuration ω of the system can be interpreted as the age of the information concerning every agent in the system (in other words the time since the last communication where an update of knowledge has been done concerning the agent, for each agent). Since the error at some time t is actually the sum of all individual errors made on the estimation of each agent, and due to the linear behavior of the integral, we can split the error definition of equation (1.16) to get the sum of all individual errors integrated over all

possible configurations. However, for an individual agent, those configurations ω reduce to that agent only, and we can choose, for the estimation by an agent i of each agent j , $f_\omega = f_j(T)$, defined as the probability that the last update concerning agent j occurred some time T ago.

Then, the new definition of the error in a steady state is given by

$$\mathbb{E} [\bar{x} - y_i]^2 = \frac{1}{N^2} \sum_{j=1}^N \left[\int_0^\infty f_j(T) \mathbb{E} [(x_j(t) - e^{-\lambda_r T} x_j(t-T))^2] dT \right] \quad (1.17)$$

1.4.2 Discussion on the choice of the pdf

The brand new description of the error given in equation (1.17) raises a new difficulty: finding the probability density function $f_j(T)$. Indeed, finding the probability that the last update of knowledge about a given agent occurred T ago isn't an easy problem: a lot of factors have to be considered. Agent i could have a direct communication with agent j to update its knowledge about it, but communicating with another agent that knows information about j more up to date than agent i does could do the job too, or maybe not if it is worse. Defining this pdf is hard, and this is where we introduce our new assumption. We will try to approximate this pdf with another one, easier to compute. However, we have to be careful with the choice of this substitution pdf.

At the end of the day, we would like to obtain a lower bound on the error. Hence, we want to characterize a suitable pdf to approximate $f_j(T)$, in such a way that it leads to a lower bound.

Proposition 1. *Given some value $E = \int_0^\infty F'(t) \cdot err(t) dt$, where $F(t)$ is a cumulative distribution function, and $err(t)$ a positive function with a positive derivative, then for any cumulative distribution function $F^* > F$,*

$$E^* = \int_0^\infty (F^*)'(t) \cdot err(t) dt < E$$

Proof. Define $F^* = F + \Delta$, with $\Delta \geq 0$. One gets

$$\begin{aligned} E &= \int_0^\infty F'(t) \cdot err(t) dt = [F \cdot err]_0^\infty - \int_0^\infty F(t) \cdot err'(t) dt \\ E^* &= \int_0^\infty (F^*)'(t) \cdot err(t) dt = [F^* \cdot err]_0^\infty - \int_0^\infty F(t) \cdot err'(t) dt - \int_0^\infty \Delta \cdot err'(t) dt \end{aligned}$$

Given the fact that both F and F^* are CDFs, we also have

$$[F \cdot err]_0^\infty = [F^* \cdot err]_0^\infty = 1 \cdot err(\infty) - 0 \cdot err(0) = err(\infty)$$

Then, one can write

$$E^* = E - \int_0^\infty \Delta \cdot err'(t) dt$$

Since both $err'(t)$ and Δ are positive, we can conclude that

$$E^* < E$$

□

Proposition 1 is a big result in the generalization of the method to define the lower bound: its effect is represented in Fig. 1.6. This gives a perfect opportunity to make the link between the generalized method and the previous Ping approach presented in Section 1.3. Let us choose the CDF defining the information spreading from the Ping model: at each communication, the agents update their knowledge about the whole system. This can be reformulated as follows: agent i updated last its knowledge about any agent j the last time it was involved in any communication. Hence, we can use the pdf $f_j(T) = (N - 1)\lambda_c e^{-(N-1)\lambda_c T}$, defined in Section 1.3.2 for the Ping model, to perform the integral of the generalized method, and obtain the same bound as in the Ping model.

Coming back to Fig. 1.6, this CDF, and the bound obtained with it, are depicted for 5 agents in the system (in blue). Those are put in comparison with two other CDFs, and their induced bounds. The first one is the same Ping CDF, but evaluated at a time $t/2$ (in red). It is interpreted as, given a time, the probability that information has reached agent i in half that time, and is thus expected to be lower than the classical Ping CDF. The other one (in yellow) is simply the probability that, at any time, all agents know everything. It is translated by a probability 1 to know the state of agent j at any time. This is the biggest possible CDF, and should lead to a bound of 0. Details about the computations are given in Appendix C.

We observe on the figure that a bigger CDF implies a smaller bound. This is actually the meaning of the Proposition 1. In order to improve the results we obtained for the Ping model, we will need to find a CDF which is smaller than the one from the Ping model, but bigger than the real one, so that it is indeed a lower bound.

We can go further by saying that this gives us some freedom in the possibilities that can be tested. Provided that the chosen pdf is suitable for the method, many assumptions on the spreading mechanism can be tested to find the best possible trade-off between accuracy and complexity of the computations, which is the objective of the method.

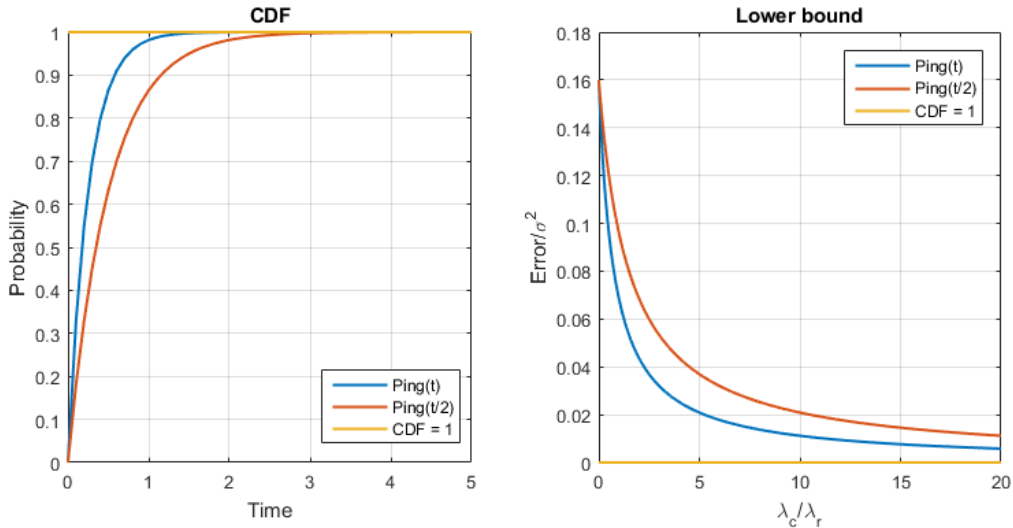


Figure 1.6: Representation of some CDFs that can be chosen to derive the pdf $f_j(T)$, and of the bound they imply. Those were obtained in a system of 5 agents. The CDF is represented for a replacement rate $\lambda_c = 1$. The blue curves stand for the CDF used in the Ping model, the red one for that CDF evaluated at a time $t/2$, and the yellow one for the case where agents always know everything.

1.4.3 Results: Infection model

Now that we settled a more general way to define a bound, we can test it for a new assumption on the pdf. Provided that the CDF related is smaller than the one from the Ping model, in accordance with Proposition 1, we expect the obtained bound to be more accurate (i.e. higher).

The assumption we will make here is the following: we try to estimate the probability that the last update of information about a given agent j happened T ago. Another way to formulate it is the probability that it takes a time T for an information coming from agent j to reach agent i in a network. The complicated part in this probability is the impact of replacements: information could be outdated or lost. We can relax this difficulty by considering there is no replacement anymore in this process. Without replacement, the time required for information to reach agent i is expected to be less, or at least as good as before. In other words, the probability that information reached agent i in a time lapse T is bigger in that case than in the real one, since we removed an obstacle of the process. Hence, if we use the pdf related to this probability in the computation of the steady state of the bound, we expect the obtained error to be a lower bound on the real one.

We call this model *Infection model*, due to the similarity between the assumption and an infection process: the probability we chose can be assimilated to the probability for an agent to be infected by information coming from another one after a given time.

Individual error

The first step in the computation of the bound is to define the individual error induced by the estimation of all other agents in the system. Let the focus be on some agent i . We want to estimate the error made on the estimation of some agent j . We define

$$\mathbb{E}_j(T) := \mathbb{E} \left[(x_j - \hat{x}_j)^2 \mid x_j(t - T) \right] = \mathbb{E} \left[(x_j(t) - e^{-\lambda_r T} x_j(t - T))^2 \right] \quad (1.18)$$

and enumerate three cases, similarly to the estimation of the error with two agents developed in Section 1.2. During the elapsed time T since the last communication between agents i and j , either i has been replaced at least once (we denote this case R_i) and has no information about j anymore, either it hasn't (N_i). In this second case, we enumerate two more cases which influence the error: either j has been replaced at least once (R_j), either it hasn't (N_j). Those represent all possible situations that could happen in T , and from then we can build the individual error for a given T .

In the following reasoning, denote by X the random variable defining the intrinsic value of agent j if it has been replaced. By definition, X , x_i and x_j are uncorrelated, and we have $\mathbb{E}X = 0$ and $\mathbb{E}X^2 = \sigma^2$. We also define once more $p_N := e^{-\lambda_r T}$ to lighten the notations.

- $\mathbb{E} [(x_j(t) - \hat{x}_j(t))^2 \mid R_i] = \mathbb{E} [x_j(t)]^2 = \sigma^2$
- $\mathbb{E} [(x_j(t) - \hat{x}_j(t))^2 \mid N_i \cap R_j] = \mathbb{E} [x_j(t) - x_j(t - T)p_N]^2 = (1 + p_N^2)\sigma^2$
- $\mathbb{E} [(x_j(t) - \hat{x}_j(t))^2 \mid N_i \cap N_j] = \mathbb{E} [(1 - p_N)x_j(t - T)]^2 = (1 - p_N)^2\sigma^2$

Then, we obtain the individual error by summing up all three cases, weighted by their probability to happen. This probability is given by the exponential law defining the replacements. We obtain

$$\begin{aligned}
\mathbb{E}_j(T) &= (1 - p_N) \mathbb{E} \left[(x_j - \hat{x}_j)^2 | R_i \right] \\
&\quad + p_N \left((1 - p_N) \mathbb{E} \left[(x_j - \hat{x}_j)^2 | N_i \cap R_j \right] + p_N \mathbb{E} \left[(x_j - \hat{x}_j)^2 | N_i \cap N_j \right] \right) \\
&= (1 - p_N^3) \sigma^2
\end{aligned}$$

Definition of the pdf

The next step in the establishment of the bound is the definition of the pdf. We described the corresponding CDF above, as the probability that some information from an agent takes a time T to reach another given agent. We can build this CDF that we will denote $P(t)$.

Denote $P_k(t)$ the probability that k agents in the system are infected by the information coming from agent j , and $P_k^i(t)$ the probability that agent i is infected knowing that k agents are infected in the system. Then, the probability that a given agent i is infected is given by

$$P(t) = \sum_{k=1}^N P_k(t) \cdot P_k^i(t) \quad (1.19)$$

One can easily obtain $P_k^i(t)$ with a little bit of combinatorial analysis.

$$\begin{aligned}
P_k^i(t) &= \frac{\# \text{ cases where } i, j, \text{ and } k-2 \text{ other agents are infected}}{\# \text{ cases where } j \text{ and } k-1 \text{ other agents are infected}} = \frac{C_{N-2}^{k-2}}{C_{N-1}^{k-1}} \\
&= \frac{(N-2)!}{(k-2)!(N-k)!} \frac{((k-1)!(N-k)!)}{(N-1)!} = \frac{k-1}{N-1}
\end{aligned}$$

Then, we have to define $P_k(t)$. We proceed as follows. Denote $I(t)$ the number of infected agents in the system, and δt a small time variation. We assume δt is small enough so that at most one more agent gets infected in the system in that time lapse. We have

$$\begin{aligned}
P_k(t + \delta t) &= \sum_{j < k} P_k(t + \delta t | I(t) = j) \cdot P_j(t) \\
&\quad \downarrow \delta t \text{ is small enough to neglect } j < k-1 \\
&= P_k(t + \delta t | I(t) = k-1) \cdot P_{k-1}(t) + P_k(t + \delta t | I(t) = k) \cdot P_k(t) \\
&= P_{k-1}(t)(k-1)(N-k+1)\lambda_c \delta t + P_k(t)(1 - k(N-k)\lambda_c \delta t) \\
\frac{P_k(t + \delta t) - P_k(t)}{\delta t} &= P_{k-1}(t)(k-1)(N-k+1)\lambda_c - P_k(t)k(N-k)\lambda_c \\
&\quad \downarrow \text{Passing to the limit } \delta t \rightarrow 0 \\
\dot{P}_k(t) &= (k-1)(N-k+1)\lambda_c P_{k-1}(t) - k(N-k)\lambda_c P_k(t)
\end{aligned}$$

This gives a system of N ODEs to solve in order to obtain the CDF

$$P(t) = \sum_{k=1}^N \frac{k-1}{N-1} P_k(t) \quad (1.20)$$

with

- $I(0) = 1 \Rightarrow P_k(0) = \begin{cases} 1 & , k = 1 \\ 0 & , \text{else} \end{cases}$
- $I(\infty) = N \Rightarrow P_k(\infty) = \begin{cases} 1 & , k = N \\ 0 & , \text{else} \end{cases}$

By definition, this CDF is supposed to be higher than the one we are actually looking for according to the definition of the error, and is thus expected to lead to a lower bound on the error, according to Proposition 1. However, we also expect it to improve the first bound derived in Section 1.3.2 with the Ping model. Still according to the same Proposition, this is achieved if the CDF is smaller than the one from the Ping model. It is verified for some numbers of agents in Fig. 1.7. The CDFs of the Infection model were obtained by numerical integration of the ODEs using *ode45* on Matlab. From this figure, it is expected to get the same bound for 2 agents, then to improve it as the number of agents in the system increases, and eventually to slowly reduce to the same bound as N gets really big.

Indeed, when $N = 2$, the Ping and Infection models are the same, since the probability for information coming from j to reach i is only governed by a communication between those two agents, as for the Ping model with two agents. Then, the CDF of the Infection model shows itself to be lower (and then better) than the one of the Ping model, until this improvement reduces while still being effective. This is expected in the sense that as N grows, the estimation of the average is expected to get better and better for both models, leading to a smaller error, and then making the improvement less apparent.

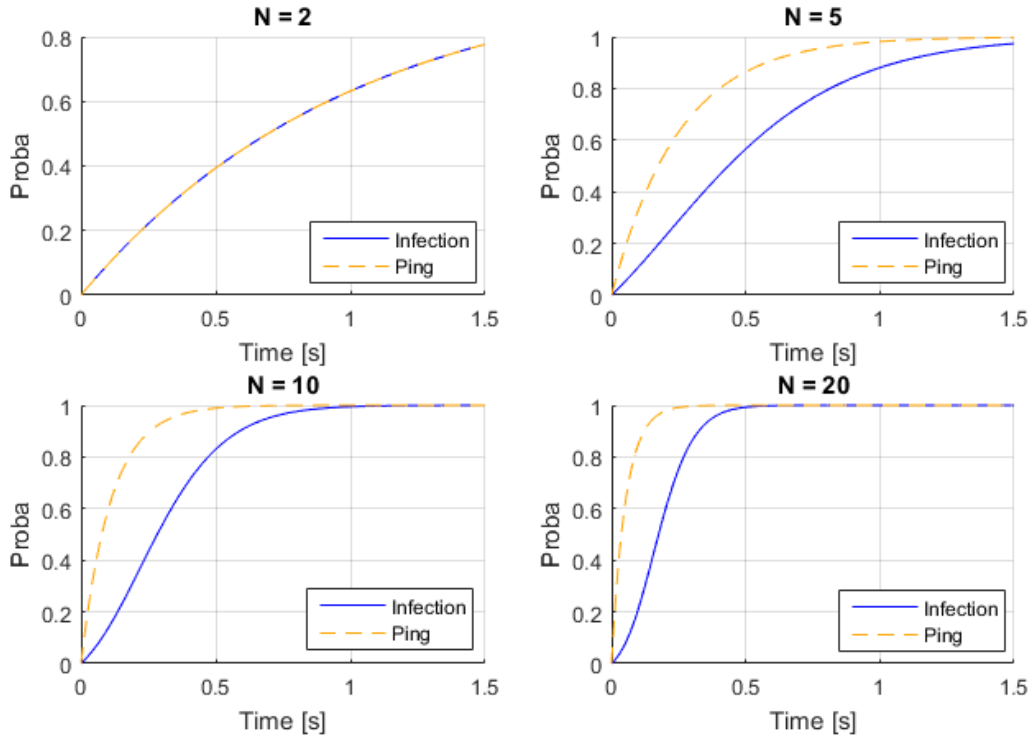


Figure 1.7: Comparison of the CDFs related to the Ping and Infection models. A higher CDF is expected to lead a lower bound. The CDFs of the Infection model were obtained by solving the ODEs with *ode45* on Matlab.

Results

Finding a way to efficiently compute this bound was a very long part of that work. Indeed, if computing the CDF, and then the bound, for a given number of agents N only requires to solve a rather simple system of ODEs, it gets hard when one wants to generalize the result to any N . The complexity of this computations relies in the fact that the ODE system changes of size with the number of agents in the system, and that the structure of the solution changes too: we introduce eigenvalues of multiplicity greater than 1, which lead to a less predictable solution for a varying N .

Still, the CDF was computed by hand for $N = 2, 3, 4$, and so has been the bound. Due to the heavy computations it requires, details are presented in Appendix D. One could keep going and compute them once and for all. However, finding a way to automatize this computation for any given N is the real goal to achieve in this part of the work. Several ideas have been tested to reach it, going from trying to establish a general form depending of N of the resolution presented in Appendix D, to an eigenvector decomposition of the solution approach, but without more success.

Once the pdf $f_j(T)$ is obtained, the bound can be computed as

$$\mathbb{E} [\bar{x} - y_1]^2 = \frac{1}{N^2} \sum_{j=1}^N \left[\int_0^\infty f_j(T) \left(1 - e^{-3\lambda_r T}\right) dt \right] \quad (1.21)$$

Since all agents are similar in the system, the expected error we make by estimating them is expected to be the same for every agent. The only error differing from the others is the one made by agent i about itself, which is always 0. Hence, since we expect to sum up $N - 1$ times the same error, one can write the above expression as

$$\mathbb{E} [\bar{x} - y_1]^2 = \frac{N-1}{N^2} \int_0^\infty f_j(T) \left(1 - e^{-3\lambda_r T}\right) dt \quad (1.22)$$

Now, we can finally compute the bound for the Infection model, for N going from 2 to 4.

- $N = 2$

$$\begin{aligned} P(t) &= 1 - e^{-\lambda_c t} \Rightarrow f_j(t) = \lambda_c e^{-\lambda_c t} \\ \Rightarrow \mathbb{E} [x_j - \hat{x}_j]^2 &= \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r}} \right) \sigma^2 \\ \Rightarrow \mathbb{E} [\bar{x} - y_i]^2 &= \frac{N-1}{N^2} \mathbb{E} [x_j - \hat{x}_j]^2 = \frac{1}{4} \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r}} \right) \sigma^2 \end{aligned}$$

- $N = 3$

$$\begin{aligned} P(t) &= 1 - (1 + \lambda_c t) e^{-2\lambda_c t} \Rightarrow f_j(t) = \lambda_c (2\lambda_c t + 1) e^{-2\lambda_c t} \\ \Rightarrow \mathbb{E} [x_j - \hat{x}_j]^2 &= \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r} \left(1 + \frac{\lambda_c}{3(\lambda_c + \lambda_r)}\right)} \right) \sigma^2 \\ \Rightarrow \mathbb{E} [\bar{x} - y_i]^2 &= \frac{N-1}{N^2} \mathbb{E} [x_j - \hat{x}_j]^2 = \frac{2}{9} \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r} \left(1 + \frac{\lambda_c}{3(\lambda_c + \lambda_r)}\right)} \right) \sigma^2 \end{aligned}$$

- $N = 4$

$$\begin{aligned}
P(t) &= 1 + (1 - 4\lambda_c t)e^{-3\lambda_c t} - 2e^{-4\lambda_c t} \Rightarrow f_j(t) = \lambda_c(12\lambda_c t - 7 + 8e^{-\lambda_c t})e^{-3\lambda_c t} \\
\Rightarrow \mathbb{E}[x_j - \hat{x}_j]^2 &= \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r} \left(1 + \frac{14\lambda_c^2 + 6\lambda_c \lambda_r}{22\lambda_c^2 + 27\lambda_c \lambda_r + 9\lambda_r^2} \right)} \right) \sigma^2 \\
\Rightarrow \mathbb{E}[\bar{x} - y_i]^2 &= \frac{N-1}{N^2} \mathbb{E}[x_j - \hat{x}_j]^2 = \frac{3}{16} \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r} \left(1 + \frac{14\lambda_c^2 + 6\lambda_c \lambda_r}{22\lambda_c^2 + 27\lambda_c \lambda_r + 9\lambda_r^2} \right)} \right) \sigma^2
\end{aligned}$$

As those results show, we are able to produce bounds, but at the cost of cumbersome computations. This illustrates why the lack of an automatizing of those computations is regrettable. Still, we have bounds that we can analyze, and the question is to see if the improvement they give in comparison with the Ping model is worth the effort.

1.4.4 Discussion on the Infection model

Correctness of the CDFs

In a first time, since the computations of the bound have been obtained by solving by hand the ODEs describing the CDFs, it is convenient to ensure the correctness of those CDFs. As those computations are quite heavy, even for a small number of agents N , a double check on the correctness of the result with the numerical solver is never a waste of time. In Fig. 1.8 are put side by side the CDF obtained by hand computations, and with the solver *ode45* on Matlab. This first check ensures us that the computations of the ODEs by hand seem valid.

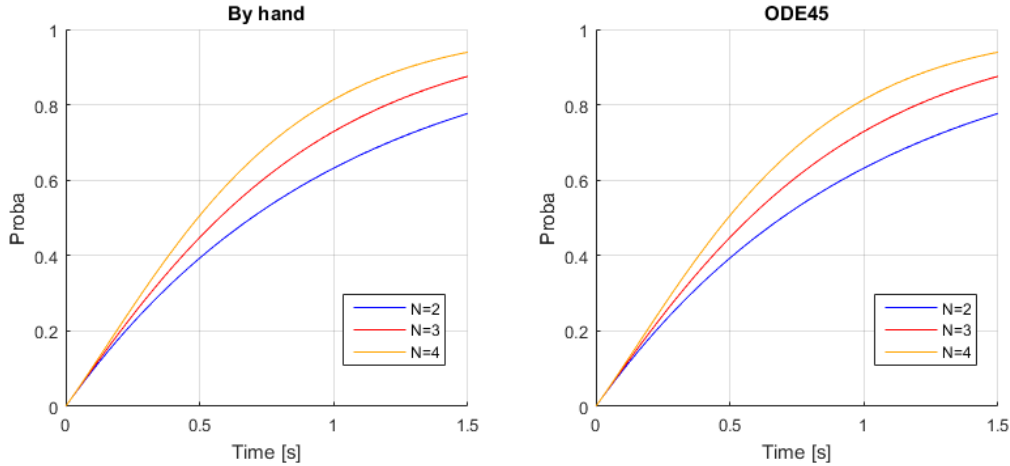


Figure 1.8: Comparison of the CDFs obtained by solving by hand the system of ODEs defined in Section 1.4.3, and with the numerical solver *ode45* on Matlab.

Sanity checks

Second, the same way we did with the bounds obtained with the Ping model, it is worth checking the consistency of the bounds through some sanity checks, in order to ensure the consistent general behavior of the bound. One more time, passing those checks doesn't ensure the validity of the bounds, but it is mandatory for them to be acceptable. The behavior of the bounds for $N = 2, 3, 4$ is depicted in Fig. 1.9. From this, let us proceed to those checks.

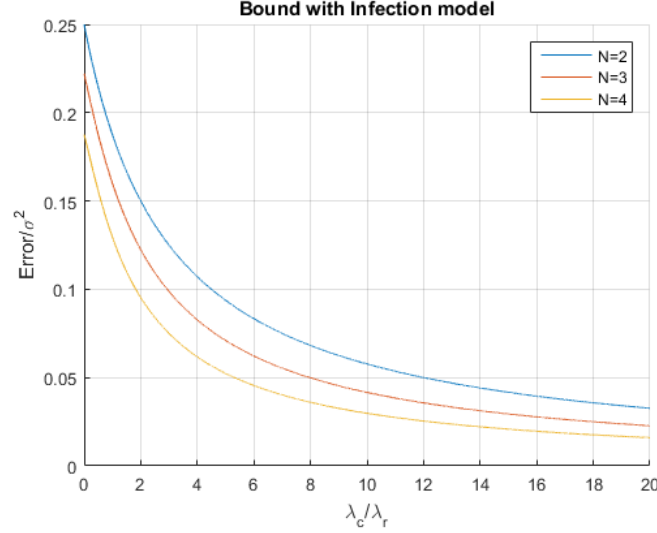


Figure 1.9: Bounds obtained for the Infection model, by solving by hand the system of ODEs defining the pdf required, computed for $N = 2, 3, 4$.

- $\lambda_c \gg \lambda_r$: when communications significantly overtake replacements, we expect agents to perfectly know the system at a steady state, and thus the error to converge to 0.
- $\lambda_c \ll \lambda_r$: inversely, when communications are overtaken by replacements, agents make the error they would when not having any information about the system, and thus by estimating every other agents with the mean, which gives an error $\mathbb{E}[\bar{x} - y_i]^2 = \frac{N-1}{N^2} \mathbb{E}[X_i^2] = \frac{N-1}{N^2} \sigma^2$, which is exactly the value of the bound when $\lambda_c/\lambda_r = 0$, no matter N .
- N increases: when more agents are in the system, we expect information to travel more easily, and thus the error to decrease.

Those behaviors are met by the bounds we obtained using this method. Hence, we can be rather confident about their validity, even though, as said above, it isn't fully guaranteed by those checks.

Improvement of the bound

One last important thing to be discussed for this model, which is one of the big motivations that led to develop it, is the improvement it gives in comparison with the Ping model. We put aside both CDFs and bounds, from both models for $N = 2, 3, 4$ in Fig. 1.10. It is expected that the Infection model improves the bounds (i.e. obtains a higher bound) as soon as $N > 2$.

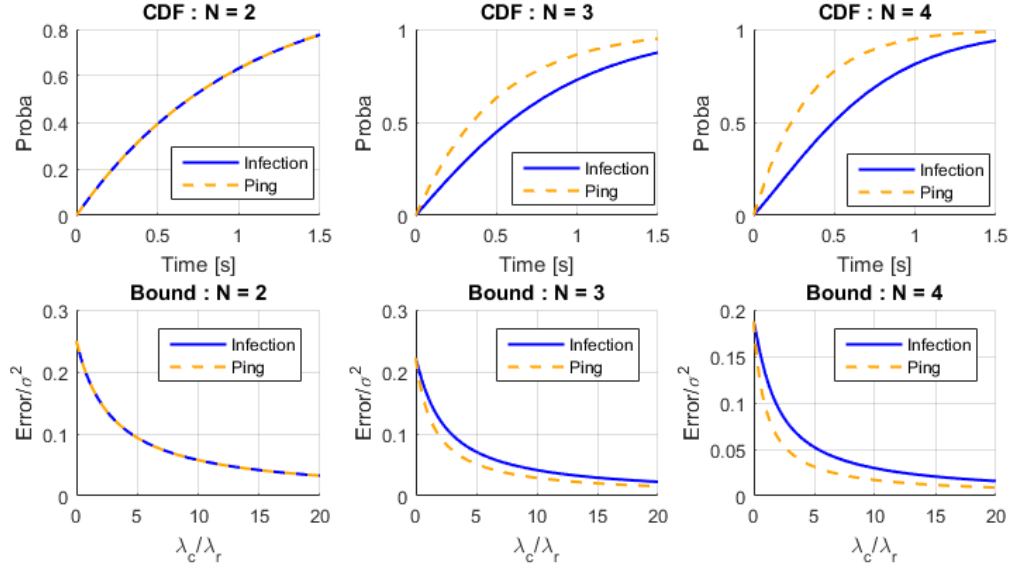


Figure 1.10: Comparison of the CDFs used for the Ping and Infection models, and the error each one of them induce, for 2, 3 and 4 agents in the system.

First things first, as predicted by the observation of the CDF, when $N = 2$, both models lead to the same bound on the error. As explained once before, this is translated by the fact that both assumptions are actually the same when $N = 2$. This first observation is a good start to assume the method works as it should.

Then, as N increases, the Infection model starts to produce a better bound on the error. This highlights several things. First, as N grows, the improvement becomes more efficient. However, the few cases of N we were able to test don't allow us to predict such a behavior forever. We actually expect this improvement to slowly decrease starting at some bigger N . Indeed as N will grow, the error will be less and less, until both errors produced by the two methods eventually tend to 0. Second, this goes in accordance with the improvement of the CDF. Indeed, the more the CDF of the Infection model is below the one from the Ping model, the higher is the corresponding bound compared to the one from the corresponding Ping model.

All those observations tend to confirm the quality of the bound. However, it is a shame that we didn't manage to establish an easy way to produce it as N grows. Still, this method is encouraging, especially in the sense that for small values of N , the improvement seems to be significant in comparison with the Ping model. The best way to improve it would be by finding a more suitable pdf for the computation of the steady state, or to obtain some efficient way to compute the current pdf for any N .

1.5 A word on the assumptions

At the end of the Chapter, we managed to develop a rather general method, which produces a lower bound on the error of estimation of the average of the system, given some assumption on the information spread mechanism. As shown in the previous parts of this Chapter, the bounds we managed to obtain were satisfactory.

However, all this work relied, as precised in Section 1.1.1, on the assumption that agents know many inside properties of the system such as the update and replacement mechanisms, or

the distribution of their intrinsic values. We already mentioned that those assumptions are way stronger than they can seem. However, let us discuss a bit more precisely about this.

In particular, the methods based themselves on the assumption that agents know the mean μ of the distribution defining their intrinsic values. It may seem to be a light assumption, but what if agents just decided to approximate the average as the mean of the distribution (this comes back to estimate the value of any agent as being μ). Keep in mind that all agents are uncorrelated.

$$\mathbb{E} [\bar{x} - y_i]^2 = \mathbb{E} \left[\frac{\sum_{j=1 \neq i}^N (x_j - \mu)}{N} \right]^2 = \frac{1}{N^2} \sum_{j=1 \neq i}^N \mathbb{E} [x_j - \mu]^2 = \frac{N-1}{N^2} \sigma^2 \quad (1.23)$$

We show the induced error in Fig. 1.11. Note that here, since no more assumptions than knowing μ and σ^2 are made to define the error, it is exactly the expected error obtained by the estimation with the mean, and not a lower bound in this case. Moreover, notice that the derived errors are still above the corresponding lower bounds (in black dotted-line on the figure) computed before with the Ping model. Indeed, keep in mind that we are no more trying to establish a lower bound here, but evaluating an arbitrary algorithm to estimate the average.

At the first sight, when N is small, this error is significant, since it is actually the worst case of all the bounds we computed. However, as N grows, even though it is still the worst possible error depicted by the bound, it gets acceptable. This is related to the fact that the bounds get more and more flat as the number of agents increases. This means that having an error on the worst value of the bound comes to be approximately on the bound, and thus to be optimal, as it flattens. In this situation, choosing μ as estimation becomes the best possible method, and as N grows, this choice is sufficient to ultimately have an error converging to 0.

This behavior shows how much the assumption on knowing the distribution is strong. In practice, we can rarely assume such knowledge for the agents. This doesn't mean that the bounds we developed are useless, but as N increases, it will be necessary to be really careful with this assumption.

Thus, an interesting extension of this work would be to consider the mean as unknown in the process. In such a situation, agents would probably try to build an estimation as accurate as possible of the mean μ to finally build their estimation of the average. This would probably lead to some additional error on that estimation. This approach will be partially mentioned in the next Chapter, since it treats the estimation of the mean μ .

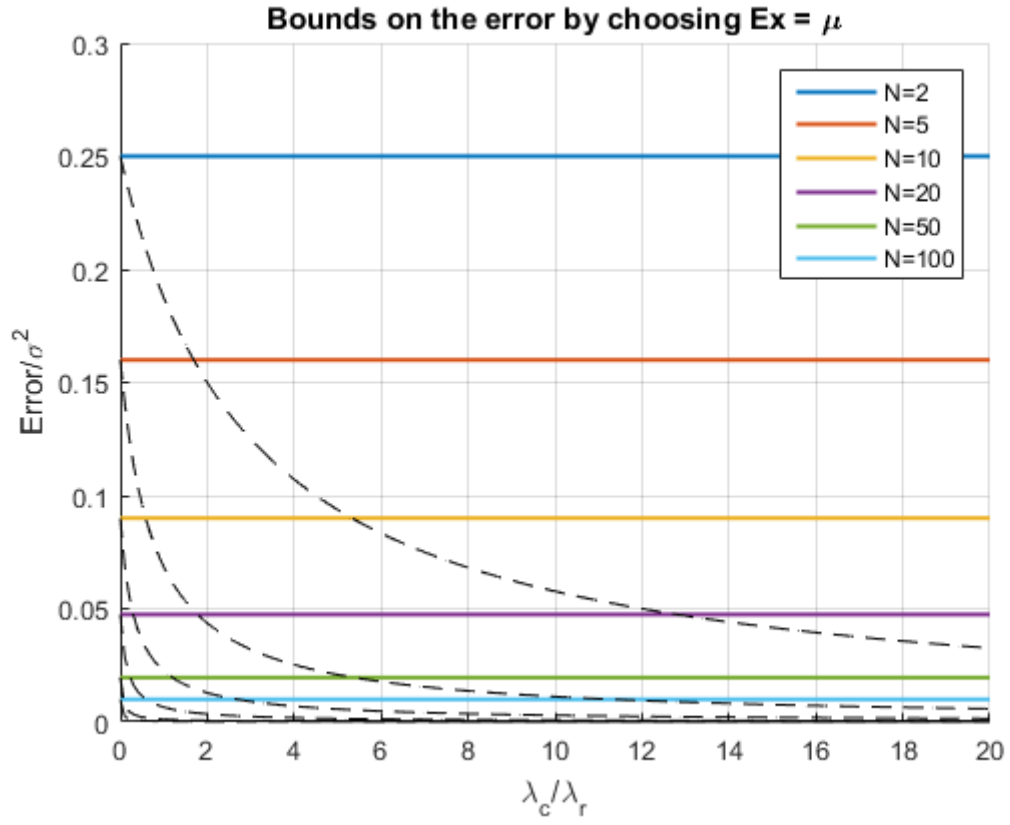


Figure 1.11: Errors obtained by choosing as estimates $y_i = \mu$ (or similarly $\mathbb{E}x_j = \mu$) compared with the corresponding Ping bound (in black dotted-lines) for several values of the number of agents in the system N .

Chapter 2

Mean Consensus Problem

2.1 Problem statement

The second problem we work on is the Mean Consensus Problem. Consider again an Open Multi-Agent System constituted of N agents, which are all defined by an intrinsic value x_i ($i = 1, \dots, N$) obtained from a distribution of mean μ and variance σ^2 . Since the system is open, arrivals and departures happen during the process. Then, through pairwise communications, agents attempt to build an estimation of the mean of the distribution μ , which is the objective of the consensus.

In order to study that problem, we place ourselves in a particular system where the intrinsic values of the agents are identically and independently distributed (i.i.d.) according to some distribution of constant mean μ and constant variance σ^2 . In this system, only replacements occur, according to a Poisson process of parameter λ_r , which represents the average number of replacements of an agent per unit of time. Similarly, the pairwise communications follow another Poisson process of parameter λ_c denoting the average number of communications between a given pair of agents per a unit of time. Such a particular system in this consensus problem is depicted in Fig. 2.1.

Due to the open character of the system, agents are expected to produce an error as they estimate the mean. In this Chapter, we will try to establish a lower bound on this error. Ultimately, we want to get this bound in a case where the system has been running for a long time, and is thus at a steady state. To bound the error, it is first required to define it, through some quality criterion for the estimation made by an agent. We arbitrarily choose to define it as the Mean Square Error on the true mean.

2.1.1 Assumptions on knowledge

Just like for the Average Consensus Problem studied in Chapter 1, we have to make assumptions on the knowledge agents have access to when computing an estimation of the mean, in order to derive a lower bound on the error. As a reminder, the more information the agents have access to, the more accurate their estimation, and thus the lower the error induced by that estimation. Hence, we will provide the agents with more information they would actually have in a real-life situation, so that the estimation they can make with a perfectly designed algorithm based on this information is necessarily better (or at least as good) than what they would have obtained with any real-life algorithm. Considering Fig. 3, such an ideal algorithm lies in the region of the improved algorithms (in yellow), which are by definition better than any algorithm in the region of the real ones (in blue), given that it is perfect with its information.

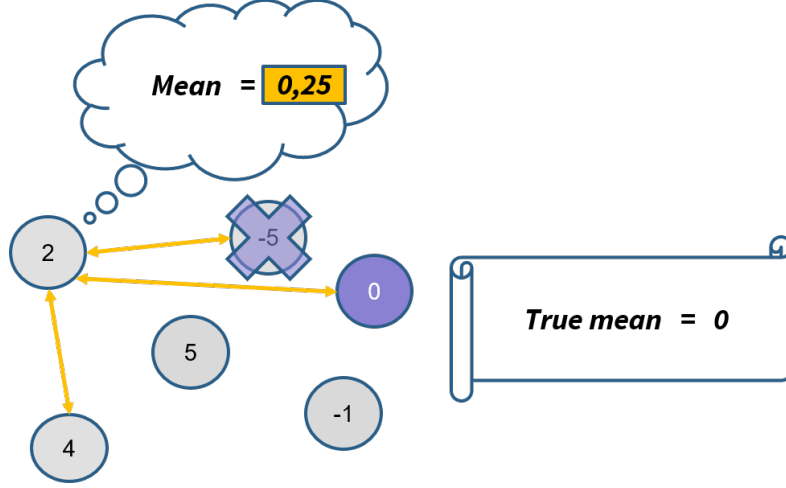


Figure 2.1: Small example of an Open Multi-Agent System of 5 agents, where replacements occur with a rate λ_r (in purple), and pairwise communications with a rate λ_c (in yellow), and where agents attempt to estimate the mean defining their intrinsic value μ .

Thus, in what follows, we will consider that agents know the size of the system N , and the fact that it is constant. Furthermore, we consider they perfectly know how replacements and communications are defined according to Poisson processes, and the corresponding rates λ_r and λ_c . The major difference with the Average Consensus Problem presented in Chapter 1 is that this time, agents only know the variance σ^2 of the distribution defining their intrinsic value, but not the mean μ they try to estimate, even though they know it is constant. In virtue with what has been explained right above, as such knowledge represents more information than what agents have in a real-life situation, we expect to effectively obtain a lower bound on the error at the end of the day with such assumptions.

2.1.2 Definition of the estimator and the error

Under this improved knowledge agents have, in order to obtain the desired lower bound, agents have to estimate the mean μ with an ideal algorithm based on it. An algorithm is defined as ideal if the estimation it provides minimizes the error of estimation (that we defined as the Mean Square Error). Hence, the estimation μ_i of μ performed by agent i is the Minimum Square Error estimator defined as follows.

$$\begin{aligned}
 \mu_i &= \arg \min \left(\mathbb{E} \left[(\mu - \mu_i)^2 \mid \text{Knowledge of agent } i \right] \right) \\
 \Leftrightarrow 0 &= \frac{d}{d\mu_i} \left(\mathbb{E} \left[(\mu - \mu_i)^2 \mid \text{Knowledge of agent } i \right] \right) \\
 \Leftrightarrow 0 &= 2\mu_i - 2\mathbb{E}[\mu \mid \text{Knowledge of agent } i] \\
 \Leftrightarrow \mu_i &= \mathbb{E}[\mu \mid \text{Knowledge of agent } i]
 \end{aligned}$$

Then, the error we would like to bound is given by the Mean Square Error definition as presented in equation (2.1).

$$\mathbb{E}[\mu - \mathbb{E}[\mu \mid \text{Knowledge of agent } i]]^2 \quad (2.1)$$

2.2 Definition of the bound

The aim of the Section will be to investigate more in depth the expression of the error of estimation of the mean $\mathbb{E}[\mu - \mu_i]^2$ defined in equation (2.1) where μ_i stands for the estimate of μ made by the agent i . For this purpose, we will first detail the expression of this estimation, by describing the ideal algorithm producing it. By doing it, we can obtain some exploitable expression of the error, that can allow us to properly define a lower bound on it.

2.2.1 Description of the ideal algorithm

In opposition with the Average Consensus Problem, the objective pursued by the agents in the Mean Consensus Problem doesn't change as agents enter or leave the system. Hence, in order to estimate μ , not only the information from the agents currently in the system can be used, but information about all the agents that already left it is relevant too. This means that information never gets outdated for the sake of the estimation, and deserves to be kept in memory by the ideal algorithm.

Then, to estimate μ , an agent i relies on the information it has about every agent it knows that have passed through the system since it started running. To picture this, figure that each agent owns some *Acquaintance repertory* that contains the label and corresponding information of all the agents that it knows have lied in the system someday. From this, we define $n(t)$, the size of the Acquaintance repertory of an agent i at some time t , that also designs the amount of information used by agent i to build its estimation of μ . This quantity $n(t)$ is a strictly positive integer random variable, which takes by default the value 1 for an agent entering the system (since it only knows itself at that time). Through communications with agents in the system, it will update its repertory with information owned by the other agents. This information could concern an agent that left the system. An example of such a description is depicted in Fig. 2.2.

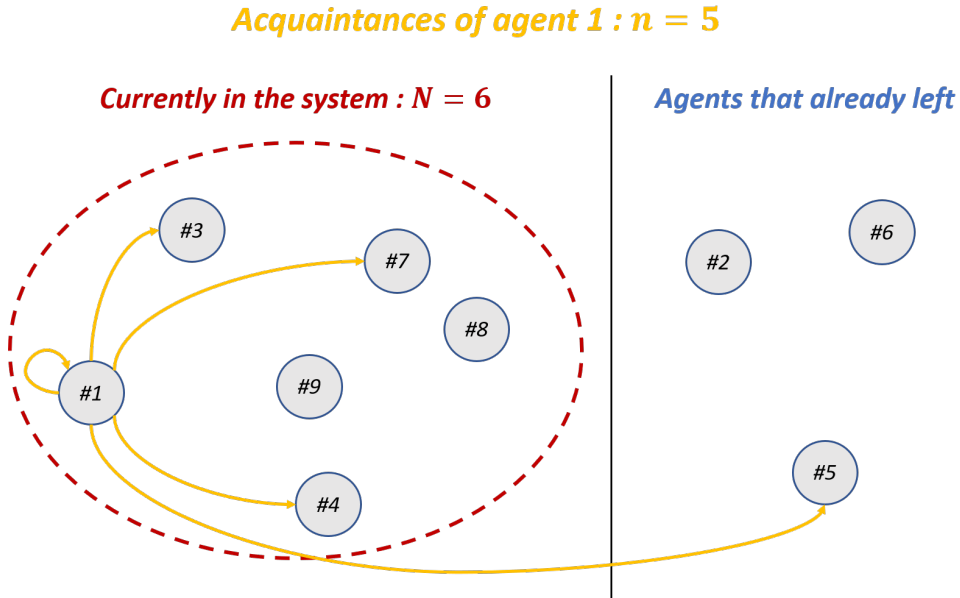


Figure 2.2: Example of a system of 6 agents trying to solve a Mean Consensus Problem by using an Acquaintance repertory. A yellow arrow represents the fact that the emitter of the arrow knows the destination. In this example, agent 1 knows five agents: itself, agents 3, 4 and 7 that currently are in the system, and agent 5 that already left it.

Then, we can develop the expression of the estimation of the mean made by agent i . Let us denote by X_j ($j = 1, \dots, n$) the intrinsic value of the j -th agent listed in the Acquaintance repertory of the agent i . Then, its estimation μ_i is entirely build, through the ideal algorithm, based on this information.

$$\mu_i(t) = \mathbb{E}[\mu \mid \text{Knowledge of agent } i \text{ at time } t] = \frac{1}{n(t)} \sum_{j=1}^n X_j \quad (2.2)$$

2.2.2 Expression for the bound

From the expression of the estimation obtained in equation (2.2), we can develop more in details the expression error made by agent i , given the size of its Acquaintance repertory n .

$$\begin{aligned} \mathbb{E}[(\mu - \mu_i)^2 \mid n] &= \mathbb{E}\left[\mu - \frac{1}{n} \sum_{j=1}^n X_j\right]^2 = \mathbb{E}\left[\frac{\sum_{j=1}^n (\mu - X_j)}{n}\right]^2 \\ &= \frac{1}{n^2} \mathbb{E}\left[\sum_{j=1}^n (\mu - X_j)^2 + \sum_{j \neq k} (\mu - X_j)(\mu - X_k)\right] \\ &= \frac{1}{n^2} \sum_{j=1}^n \mathbb{E}[(\mu - X_j)^2] + \frac{1}{n^2} \sum_{j \neq k} \mathbb{E}[(\mu - X_j)(\mu - X_k)] \\ &= \frac{1}{n^2} n \sigma^2 + \frac{1}{n^2} \sum_{j \neq k} (\mathbb{E}[\mu^2] - \mathbb{E}[\mu X_j] - \mathbb{E}[\mu X_k] + \mathbb{E}[X_j X_k]) \\ &\quad \left\{ \begin{array}{l} \text{Since agents are uncorrelated: } \mathbb{E}[X_j X_k] = \mathbb{E}[X_j] \mathbb{E}[X_k] \end{array} \right. \\ &= \frac{\sigma^2}{n} + \frac{1}{n^2} \sum_{j \neq k} (\mu^2 - \mu^2 - \mu^2 + \mu^2) = \frac{\sigma^2}{n} \end{aligned}$$

This is the exact definition of the error produced by an agent i when estimating the mean μ , given the quantity of information it has to build its estimation. In order to determine the expected error for any value of n , we choose some random agent in the system i . Since all agents are interchangeable, this can be done without loss of generality. Then, to get the expected error in a steady state, we have to compute the expected value of the above expression for the error, provided that the system has been running for a long time to reach a steady state.

$$\mathbb{E}[\mu - \mu_i]^2 = \mathbb{E}\left[\frac{\sigma^2}{n}\right] = \mathbb{E}[1/n] \sigma^2 \quad (2.3)$$

Hence, in order to evaluate the expected error made by agent i when estimating the mean μ at time t , one wants to compute:

$$\mathbb{E}[1/n] = \sum_{k=1}^{\infty} \frac{1}{k} P(n = k) \quad (2.4)$$

Computing the expected value $\mathbb{E}[1/n]$ would provide an exact estimation of the desired error under the assumptions described in Section 2.1, and thus a lower bound on the error produced by any realistic algorithm, which is what we aim for. However, this is where the computations become hard.

Indeed, to build $\mathbb{E}[1/n]$ in a steady state, we need to proceed in a similar way as we did for the Average Consensus Problem when trying to derive the expected error of estimation of the value of an agent in a steady state (in Section 1.4.1). Indeed, given some moment in time, we have to consider every possible configuration of the system, and evaluate the expected value of $\mathbb{E}[1/n]$ in each of these particular configurations. Then we can obtain the desired expected value by averaging all values of $\mathbb{E}[1/n]$ in a configuration, weighted by the probability for the system to lie in this configuration. This is really hard to do in the sense that it relies on a huge amount of things that must be considered in the computations. Indeed, at each replacement, we face the risk to loose information forever if it is not maintained through communications. Moreover, at each communication, it is hard to define what new information is gathered. Those are only two examples of difficulties to be handled when evaluating this expected value.

Hence, we will introduce a new assumption to simplify those hard computations, in such a way that it provides a lower bound on the desired error we intended to compute. Since this expression was supposed to be a lower bound on the error, the introduction of this assumption which bounds by below this expression should also lead to a lower bound on the error produced by a realistic algorithm.

Proposition 2. *Given a strictly positive random variable X , then*

$$\mathbb{E}[1/X] \geq 1/\mathbb{E}[X]$$

Proof. Jensen's inequality [11] says that for a convex function $f(x)$, a set of real values x_i ($i = 1, \dots, N$) in the domain of f , and positive values ω_i ($i = 1, \dots, N$) such that $\sum_{i=1}^N \omega_i = 1$, then

$$\sum_{i=1}^N \omega_i f(x_i) \geq f\left(\sum_{i=1}^N \omega_i x_i\right)$$

By choosing ω_i the CDF $P(X = i)$, one can then write this expression

$$\mathbb{E}[f(X)] \geq f(\mathbb{E}[X])$$

Since $f(x) = 1/x$ is convex for $x > 0$, we conclude

$$\mathbb{E}[1/X] \geq 1/\mathbb{E}[X]$$

□

Proposition 2 allows us to write

$$\mathbb{E}[\mu - \mu_i]^2 = \mathbb{E}[1/n] \sigma^2 \geq \frac{\sigma^2}{\mathbb{E}[n]} \quad (2.5)$$

Thus, $\sigma^2/\mathbb{E}[n]$ is a lower bound on the error of estimation of μ . Still, $\mathbb{E}[n]$ is also hard to compute for the same reasons as $\mathbb{E}[1/n]$. However, it has in comparison the advantage to be easy to simulate and interpret. Indeed, by building simulations of an Open Multi-Agent System trying to solve a Mean Consensus Problem the way described in this Section, we can show the behavior of $\mathbb{E}[n]$, and get then a first insight of it. However, as we will mention in the next Section, we won't be able to simulate the system in a steady state. Then, simulating the behavior of $\mathbb{E}[n]$ only provides a first impression of a corresponding problem in a transient state. Still, this is something that we can do in a first time.

Then, in the next Section, we will realize such simulations. From them, we intend to get a first impression of the behavior of $\mathbb{E}[n]$ which is used to define the bound. Then, we expect to be able to have a first insight of the behavior of this bound.

2.3 Simulations

We just defined a bound on the error of estimation of μ made by an agent i based on the expected value of the amount of information it uses to build its estimation denoted $\mathbb{E}[n]$. As we also precised right above, the computation of this expected value in a steady state of the system is hard, due to the numerous situations that need to be considered.

To deal with this complexity of the computations, a first solution can be to run several simulations that will allow us to depict the behaviors we intend to characterize. In this Section, we will present the results of such simulations on a system of the type we study. Thanks to those, we expect to be able to produce a first insight of the behavior of $\mathbb{E}[n]$. If we manage to obtain it, we can expect to extend it to the definition of the bound relying on it, and thus to have a first impression of the behavior of such bound. Such results could then be the base of potential further investigations in the establishment of a formal definition of the desired bound.

2.3.1 Perspectives

The problem we attempt to solve raises some sort of a paradox in the dependence of the quality of the estimation with the parameters defining the mechanisms ruling the system. Indeed, as the replacement rate increases, more agents pass through the system. Thus, those agents bring with them additional information, allowing to refine the estimation of the mean made by the agents as they learn it. However, as this rate grows, it also increases the risk for this information to disappear from the system if communications are not frequent enough to maintain it. Indeed, on the first hand, in a system where no replacement occur, we cannot expect to build a better estimation than the one relying on the initial agents in the system. On the other hand, in a system where replacements are way more frequent than communications so that their impact is negligible, agents never have the time to learn from other agents before being replaced again, and they are the only information they have.

Let us formalize those suppositions under the form of two cases. The above description relies on some dominant behavior of the replacements or communications. In the first situation, communications are such that replacements don't alter that much their effect, and only bring new information in the system. In the second one, it is the opposite, and replacements are such that communications progressively cannot follow the pace of departure of agents. Similarly with the Average Consensus Problem, we can express this with the ratio of both rates. However, for this problem, we will work with the inverse of the ratio we used for the Average Consensus Problem, given by λ_r/λ_c . As we will see further, depicting results like this allows a better readability of the behavior we intent to show.

- If λ_r/λ_c is small, then information brought by replacements is expected to stay in the system through communications, and the quality of the estimation is expected to grow with the number of new information agents have at their disposal.
- If λ_r/λ_c is big, then information brought by replacements is expected to be ultimately lost, so that only recent information is maintained in the system by the communications, to eventually totally erase the effects of communications.

In this two-times behavior, we expect to bring out a critical value of the ratio of the rates λ_r/λ_c , that we denote λ_{crit} , which marks the transition from one case to another. Characterizing this critical value would be a first step in the characterization of the behavior of $\mathbb{E}[n]$, and thus

by extension, of the bound on the error we attempt to evaluate.

In order to validate those suppositions, and to ultimately characterize $\mathbb{E}[n]$, we will run several simulations of Open Multi-Agent Systems in a Mean Consensus Problem. One of the aims of those will be to highlight the two-times behavior we just described, and potentially to derive some other conclusions we could make on the behavior of $\mathbb{E}[n]$.

By nature, the simulations are not expected to bring out the asymptotic behavior of the system. In other words, we cannot expect them to provide a result in a steady state. Indeed, we will need to define some starting point, and then to let the simulations run as far as we can so that we can observe what we aim to. Thus, the results that will follow stand for a system in a transient state. However, we will try to extend the observations we will make in a case when the system has been running a very long time, in order to get a (maybe rough) first impression of the steady state behavior.

2.3.2 Calibration of the simulations

Before performing the simulations, it is required to describe and calibrate them. We place ourselves in the context of a Multi-Agent Open System where N agents lie simultaneously. Replacements occur according to a Poisson process of parameter λ_r , and pairwise communications according to another Poisson process of parameter λ_c . Also, intrinsic values are distributed according to an arbitrary distribution of constant mean μ that agents tend to estimate, and of constant variance σ^2 .

Initially, N agents are present in the system. They aim to estimate, at any time, the mean of the distribution defining their intrinsic value by using the Minimum square error estimator based on their knowledge about every agent that passed through the system. At the beginning of the simulations, and each time it is replaced, an agent resets its knowledge, so that it only knows itself.

Events related to Poisson processes have a probability to happen in a time lapse δt defined according to an exponential law of the same parameter as the process. Due to the discrete nature of the simulations, we will simulate the replacements and the pairwise communications as follows. The time is discretized in time steps of constant size δt . At each step, we evaluate the probability that an event happened in the time lapse, using the exponential law. This yields a risk, in the sense that the exponential law defines the probability for an event to take place at least once. Hence, there exists a probability that the event happened more than once in the time lapse, and thus to have events collisions. This influences the accuracy of the simulations, since it is assumed to perform them that at most one occurrence of each event happened in a time step. To ensure it works correctly, we will have to ensure the probability of more than one event happening being low enough in comparison with the probability that one or none happens, under the parameter definitions we will choose for the simulation. We will come back to this when defining this time step δt .

Arbitrarily, we decide to work on a reference system of 4 agents. This case has the advantage to imply rather fast simulations, and yet to provide observable results. Agents' intrinsic values are decided to be randomly chosen uniformly in the range $[-10, 10]$. They are thus distributed according to a Uniform distribution of zero-mean, and of variance $\sigma^2 = \frac{(10+10)^2}{12} = \frac{400}{12} \approx 33.3$.

Since we intend to show the dependence of the observed behaviors of the system on the ratio of the rates, we choose as a reference case to set the replacement rate $\lambda_c = 1$. From this,

we will study the cases obtained with a replacement rate varying from 0 to at most 10. The choice of this range of rates is driven by empirical observations. Indeed, under the definition of the parameters of the simulation, this range of rates depicts the wanted behavior in an observable way. Beyond 10, the replacement rate, as we will see, has already, in most of the cases, overtaken the communications, and the observed behaviors have already become as they will be asymptotically when $\lambda_r \rightarrow \infty$.

Finally, as mentioned before, we have to ensure the validity of our choice of time step to perform the simulations. Assume the rates are defined in $[s^{-1}]$. Then, let us choose, arbitrarily, to work with a time step of size $\delta t = 0.01$ $[s]$. We will check the conditional probability for more than one event to take place, given that one happened. For this, denote $N_e(t)$ the random variable representing the number of events following a Poisson process of parameter λ that happened in a time t .

The probability for at least one event to happen in δt for a rate λ is given by the exponential law

$$P(N_e(\delta t) > 0) = 1 - e^{-\lambda \delta t} \quad (2.6)$$

Then, from the definition of the Poisson law, the probability of an event happening exactly k times in a time δt is given by

$$P(N_e(\delta t) = k) = \frac{(\lambda \delta t)^k}{k!} e^{-\lambda \delta t} \quad (2.7)$$

Hence, the probability to have more than one event is given by the following expression

$$P(N_e(\delta t) > 1) = 1 - P(N_e(\delta t) = 0) - P(N_e(\delta t) = 1) = 1 - (1 + \lambda \delta t) e^{-\lambda \delta t} \quad (2.8)$$

Then we define the conditional probability that at least two events happened in a time δt , given that at least one happened, using Bayes Theorem about conditional probabilities.

$$P(N_e(\delta t) > 1 \mid N_e(\delta t) > 0) = \frac{P(N_e(\delta t) > 1 \cap N_e(\delta t) > 0)}{P(N_e(\delta t) > 0)} = \frac{P(N_e(\delta t) > 1)}{P(N_e(\delta t) > 0)} \quad (2.9)$$

This finally gives the expression of the conditional probability we look for, which is depicted in Fig. 2.3 for the range of rates we will use in our simulations (from 0 to 10).

$$P(N_e(\delta t) > 1 \mid N_e(\delta t) > 0) = \frac{1 - (1 + \lambda \delta t) e^{-\lambda \delta t}}{1 - e^{-\lambda \delta t}} = 1 - \frac{\lambda \delta t e^{-\lambda \delta t}}{1 - e^{-\lambda \delta t}} \quad (2.10)$$

As the rate increases, the conditional probability increases too. In the range we defined, we have at worst, when $\lambda = 10$, a conditional probability around $1/20$. This probability is unfortunately a bit high to be totally neglected. However, the purpose of the following simulations is to provide a first, maybe rough, insight of the behavior of $\mathbb{E}[n]$, and of what follows from it. Furthermore, this probability is reached for big values of the rate. When the rate is lower, this probability decreases in such a way that we can expect such event to be rather rare. Hence, even though this choice reveals to be a bit imperfect, we will work with it, accepting the possible small loss of accuracy it induces.

The above discussion shows that the following simulations have some improvement margin. We could improve them, for instance, by defining some adaptive time step, that always satisfies some threshold for this conditional probability. Another way of doing it would be to work

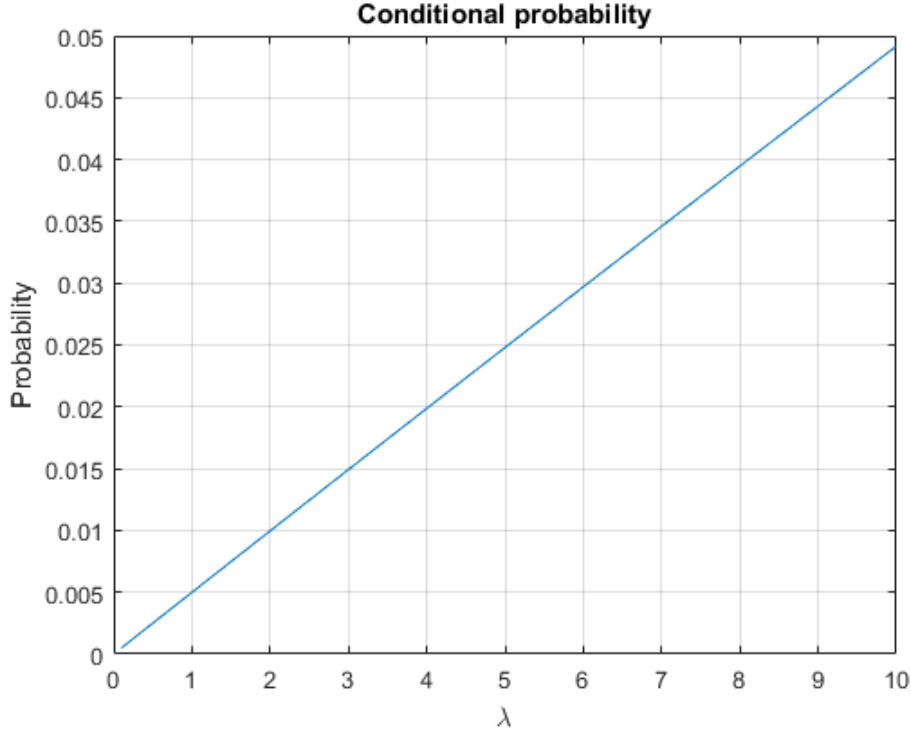


Figure 2.3: Conditional probability that an event following a Poisson process of rate λ [s^{-1}] happened at least twice, given that it happened at least once in a given time $\delta t = 0.01$ [s], for several values of the rate λ .

differently on the events definition. One could compute the random time to wait until the next event, for each event, and work that way to discretize the time.

However, the following simulations, if they are not perfectly accurate, will show themselves satisfactory, in the sense that they will allow us to make first interesting observations on the wanted behavior, that could be used for a possible extension of this work.

2.3.3 Results

We will finally show here the results obtained from the simulations described above. From them, we expect to be able to build some suppositions on the transient, and maybe steady state, behavior of $\mathbb{E}[n]$, and what follows from it.

In what follows, we choose to depict the results obtained from agent 1, after some time spent in the system. Since all agents are interchangeable, we can choose it without loss of generality. Initially, agents only know themselves. Also, simulations are reproduced 200 times and averaged to get an expected value of the results.

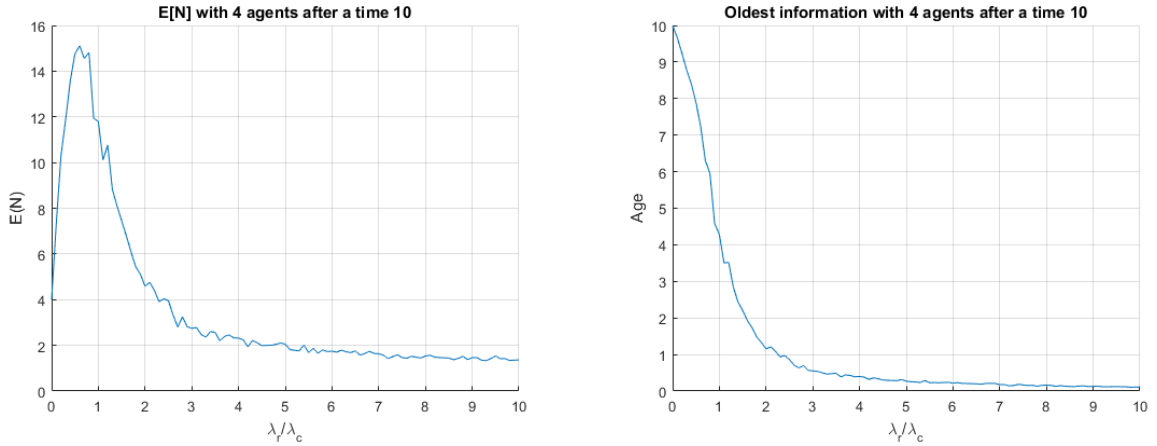
Reference case

The first simulation we run is the classical reference case, under the parametrization described in Section 2.3.2. The aim of this simulation is to depict the behavior of $\mathbb{E}[n]$ after a time of 10 [s] in Fig. 2.4a. We also depict the age of the oldest information used by the agent to build its estimate. We expect it to be correlated with the critical value of the rate ratio λ_{crit} defined before as the value of λ_r/λ_c where we expect the behavior of $\mathbb{E}[n]$ to change. Indeed, in the first time of the simulation, when information is expected to stay in the system, we expect the age of the oldest information to grow with the age of the system, as information is maintained. Once

the information starts getting lost, then we expect that age to stay more constant, in the sense that older information is expected to be lost in the process. We hope to be able to characterize λ_{crit} with this information too.

Fig. 2.4a is highly encouraging, since it shows the suspected two-times behavior, described in Section 2.3.1. Indeed, as λ_r/λ_c grows, so does $\mathbb{E}[n]$ in a first time. Then, after reaching some peak, $\mathbb{E}[n]$ starts drastically decreasing, until eventually reach 1, which is the value obtained when no communication at all occur.

Moreover, Fig. 2.4b seems to be rather correlated. Indeed, the age of the oldest information decreases with the time. However, the decrease is rather fast, and doesn't bring up some constant behavior as $\mathbb{E}[n]$ increases. Still, the decrease looks like it fastens once the assumed critical value λ_r/λ_c is passed.



(a) $\mathbb{E}[n]$ for 4 agents in the system, for several rate ratios λ_r/λ_c , under the parameters definition given for the reference case, simulated over a time lapse of 10 [s].

(b) Age of the oldest information used to build the estimation of μ for 4 agents in the system, for several rate ratios λ_r/λ_c , under the parameters definition given for the reference case, simulated over a time lapse of 10 [s].

Figure 2.4: $\mathbb{E}[n]$ and age of the oldest information used in the estimation of μ by an agent in the reference case, for a time lapse of 10 [s].

Study of the age growth

We just proposed the possibility for the age of the oldest information to be correlated with the two-times behavior of $\mathbb{E}[n]$. In Fig. 2.5, we show the evolution of that age with the time as the rate ratio increases.

In our supposition, we expect that age to show a linear behavior in the case where λ_r/λ_c is small, since information is expected to be maintained in the system along with the process. When λ_r/λ_c increases, then we expect the age to flatten, to eventually keep a constant value, when communications can be considered to be totally overtaken by the replacements. This behavior is met by the curves presented in the figure. However, the transition from the linear behavior to the constant one is rather smooth. Still, such an observation could be the base to define a criterion of quality to estimate the critical value λ_{crit} . Indeed, by defining a “critical slope” for these results, we could create a relation between that slope and the critical value of λ_r/λ_c , and maybe design an algorithm able to identify this critical value.

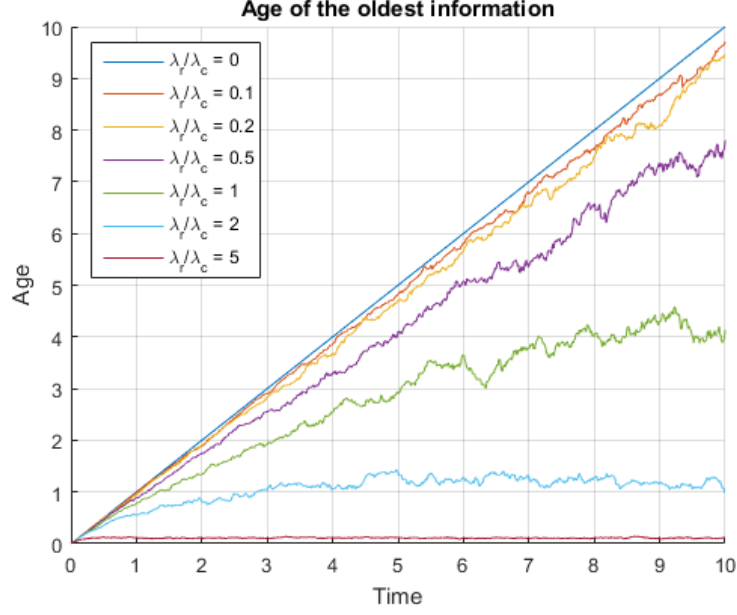


Figure 2.5: Evolution of the age of the oldest information used by an agent to build its estimation according to the time, for several values of λ_r/λ_c , in the reference parametrization.

Variation of N

For the next simulation, we will study the case where the number of agents simultaneously present in the system N changes. In Fig. 2.6 are depicted side by side $\mathbb{E}[n]$ for different values of N , and the corresponding error of estimation produced by the agent. The error is measured directly in the simulation, by computing the mean square error among all the iterations of the simulation, of the estimation of μ by agent 1 at the end of the time lapse.

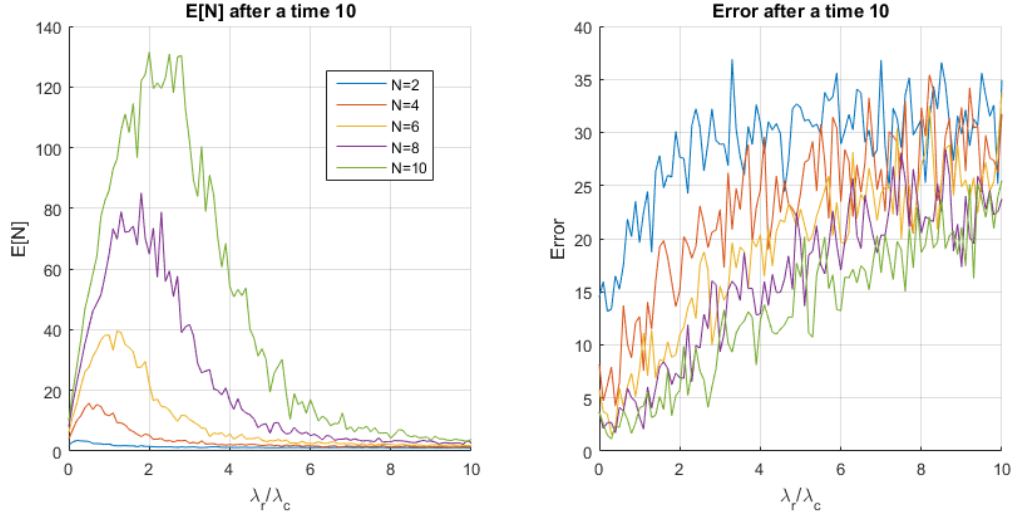


Figure 2.6: $\mathbb{E}[n]$ and corresponding error of estimation of the mean μ in the reference parametrization, when we change the number of agents simultaneously in the system N .

When N increases, we observe an increase of the expected amount of information used to build the estimation of μ . In the first place, this expected value increases, in agreement with our supposition that in a first time the system maintains the information. With more agents present

in the system, the quantity of information naturally increases, and the number of replacements (and thus incoming of new information) too. Then, when $\mathbb{E}[n]$ starts to decrease, it stays bigger for bigger N , to ultimately converge to the same value when communications are overtaken by replacements.

This is translated in the estimation of the error by a generally lower error for a bigger N . This conclusion was also verified for the Average Consensus Problem, and is consistent in the sense that the more agents, the more information, but also the more opportunities to be involved in a communication, and thus to improve the estimation. Once more, the error seems to ultimately converge to the same area, as communications are overtaken by replacements, which is consistent with the lower bound on the error definition from Section 2.2.

Note also that the error slightly decreases for low value of λ_r/λ_c . This translates the increase of $\mathbb{E}[n]$ in this situation, since it is supposed to imply a better estimation. However, this will be more clear in further simulations.

Finally, let us notice that the critical value λ_{crit} , translated in the figure by the peak that reaches $\mathbb{E}[n]$ before decreasing, in opposition with what will be shown in the next simulations, tend to change as N changes. This seems to imply that the critical value depends on the number of agents present in the system. To characterize it, it may be either necessary to build some kind of algorithm to determine it for a given value of N , or to find the relationship between λ_{crit} and N .

Different time lapses

As precised before, the simulations performed here stand for the transient state of the system. However, those could bring out an idea on how the system will evolve until reaching a steady state. To do so, we show in Fig. 2.7 side by side $\mathbb{E}[n]$ and the error induced by the estimation built by an agent (computed as the mean square error among all the iterations of the simulation) at several points of time between 0 and 15 seconds.

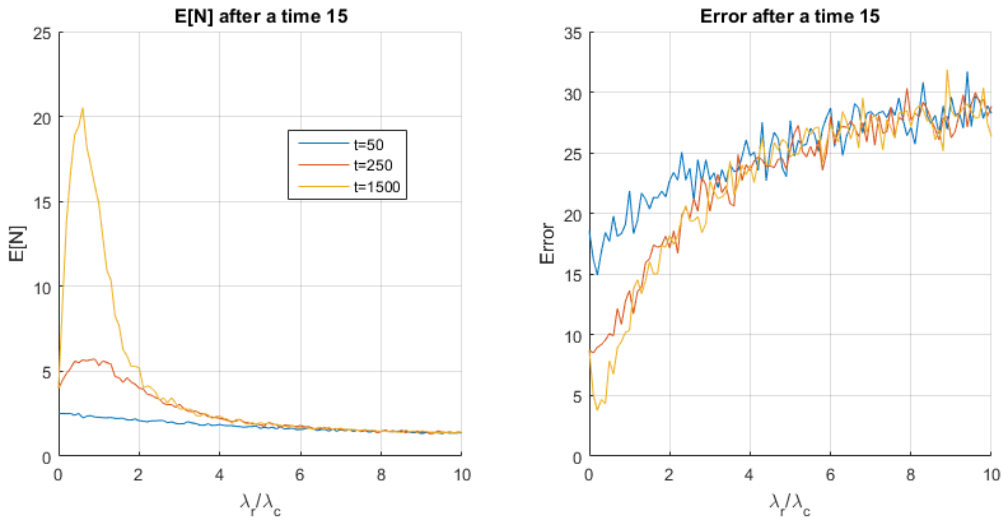


Figure 2.7: $\mathbb{E}[n]$ and corresponding error of estimation of the mean μ in the reference parametrization, represented for several time steps along the process.

The main interest of this figure is to show the effective evolution of both $\mathbb{E}[n]$ and the error as the time passes. Indeed, the more time spent in the system, the bigger $\mathbb{E}[n]$ for low values of λ_r/λ_c , when replacements are handled by communications. After that, as $\mathbb{E}[n]$ decreases, it tends to quickly reach the same curve. Similarly, as the time passes, the error induced gets lower and lower as long as λ_r/λ_c is small enough. After that, the error tends to converge to the same value as communications are overtaken by replacements. Note also that in this situation, no matter the time, λ_{crit} seems to be located at the same value.

Hence, by extending those observations, we can try to figure out suppositions about the steady state behavior of the system. We can expect the error to asymptotically go close to 0 as long as replacements don't take information away. Similarly, when λ_r/λ_c gets high enough, we expect the error to asymptotically stay on the value to which the errors converge here. Of course, those are suppositions. But still, they seem to be a good first impression of the behavior of the estimation and of the error it induces, when the system goes to its steady state.

2.3.4 Discussion

The simulations we just presented aimed to study the transient behavior of a system as the one presented in Section 2.1, in order to solve a Mean Consensus Problem. From those simulations, we expected to build a first insight on the behavior of the amount of information available for an agent to build its estimation of the mean μ , denoted $\mathbb{E}[n]$, which is used to define our bound on the error produced by this estimation. As a reminder, we defined the lower bound on the error as $\sigma^2/\mathbb{E}[n]$.

Thanks to the above results obtained through those simulations, we were able to make some suppositions about the transient behavior, and can expect to derive some intuition on the steady state of the error.

The first thing that we settled is the presence of some critical value for λ_r/λ_c which separates the evolution of $\mathbb{E}[n]$ into two cases. It first increases as the replacement rate is low enough in comparison with the communication rate to bring information that will not be lost afterwards, and then decreases to 1 as replacements imply more loss than new information. Such observation brings out a first characterization of $\mathbb{E}[n]$, and by extension of the error. If we manage to compute that critical value of λ_r/λ_c , we could expect to create separate estimations of both behaviors. Also, we can expect the steady state of the error to be also characterized by this critical value, in the sense that both cases should imply different types of errors.

Second, we showed the evolution of the error with the size of the system N , and with time. If the first one gave us the intuition that the critical value seems to depend on N , the second allowed us to think that, as the system reaches its steady state, $\mathbb{E}[n]$ and the error may behave as an asymptotic form of the currently shown behavior. More precisely, we could expect the estimation of the mean to asymptotically lead to a null error as long as the replacements (and thus the information income) are handled by the communications. Then, as the critical value of λ_r/λ_c is passed, an increasing error starts to be induced by the estimation, ultimately converging to some fixed value.

From those observations, we can try to bring up ideas on how to proceed to build the desired bound. A first idea could be to notice that the number of agents we know about is always bounded by the number of agents that have been through the system along with the process.

Such value is easy to compute, as it is the number of agents initially present in the system, to which all agents arrived through replacements are added. Then, we write

$$\mathbb{E}[n(t)] \leq N + \mathbb{E}[\text{Number of replacements}] = N + N\lambda_r t \quad (2.11)$$

By bounding $\mathbb{E}[n]$ from above, we obtain a lower bound on $1/\mathbb{E}[n]$, and thus a lower bound on the error $\mathbb{E}\left[\frac{1}{n}\right]$. Scaling with the variance of the distribution, it provides a bound on the error. This bound is depicted for the transient state of the system in Fig. 2.8



Figure 2.8: Comparison of the effective mean square error of the estimation, the definition of the bound on the error based on the simulation, and the lower bound defined by the number of agents that went in the system.

The figure shows two main things. First, the effective error obtained through simulations is indeed bounded by below by $\sigma^2/\mathbb{E}[n]$. Second, the bound obtained by choosing the expected number of agents passing through the system effectively bounds the error by below. However, it shows itself to be quite useless. Indeed, it quickly goes to 0 as the replacement rate increases. Hence, the only region where it makes sense is in the small values of λ_r/λ_c . However, the observations made when the time increases tend to let us think that in this region, the error is expected to tend to 0 in the steady state (and so does the bound). Hence, this bound, even though it effectively bounds the error, reveals to be useless.

A second possibility would be to characterize the bound according to the critical value λ_r/λ_c . For that purpose, we would need a way to find that value. As presented with the simulations of the age of the oldest information used to build the estimation, one could find a criterion to determine whether we are close to this value. In this context, we could build an algorithm, for instance based on the bisection method with that criterion, in order to determine it.

At the end of the day, we could not manage to determine an exact formulation of a bound for the Mean Consensus Problem. However, we were able to formulate it according to the expected amount of information used to build the estimation. And through the above simulations, we were able to bring out a first impression on the characterization of this expected value. Further investigations in that sense may lead to more results in the determination of the desired bound for potential later work.

Conclusion

The study of Open Multi-Agent Systems is a young but ever growing research field. Nowadays, considering decentralized approaches is more and more relevant, with the growing size of the studied systems, and with the numerous advantages it provides. First of all, it allows working with big systems thanks to the distribution of information among the agents in the system. Second, it prevents entire systems to rely on central units, to which the success of an operation would be dependent. Moreover, it allows agents to keep their own approximation of the solution of a problem, making them more reactive to perturbation.

Recently, considering such systems open has become a major and necessary improvement of decentralized methods for solving problems. Open systems are getting more frequent as the size of systems grows, since it increases the probability for agents to enter or leave the system. Furthermore, some systems are by nature open. However, most of the existing work concerning decentralized approaches stands for closed systems. There is today a real need to develop a more appropriate theory to deal with open systems.

In this Master thesis, we intended to build some bases of the Open Multi-Agent Systems theory. One of the parts of the elaboration of such a theory lies in the resolution and analysis of basic problems that are susceptible to be used for more complex tasks. In this work, we considered two of those basic problems, categorized as consensus problems. In such situations, agents attempt to converge to a common state defined in the system, that can possibly vary in time. We attempted to establish fundamental performance limitations for those problems, under the form of lower bounds on the error produced by an algorithm trying to solve the problem. To do that, we needed to describe how to relax a problem, in order to obtain an error that bounds the one obtained from any real-life algorithm. The establishment of such a bound for a problem would then provide a quality criterion for any method trying to solve the problem, and would also provide the insights of a better understanding of the potential bottlenecks that we could face when dealing with those problems.

In Chapter 1, we considered the Average Consensus Problem, in which agents try to evaluate the average of the intrinsic values of the agents present in the system. To obtain a bound on the error the agents make when estimating this average, we first presented the assumptions that we made on their knowledge about the system in order to allow them to produce an ideal estimation, better than the one any realistic algorithm would provide. Once this has been done, we started computing the bounds by solving the problem with two agents. In this first situation, we managed to derive the exact error under the knowledge assumptions, and thus obtained a lower bound on the error. When passing to the general case of N agents, we had to introduce new assumptions to relax the problem, too complicated to be solved otherwise. For this purpose, we proposed two approaches. The first one relied on a strong assumption, that made possible the computation of a lower bound on the one we would have obtained by computing the exact one under the assumptions on knowledge, and which is thus also a lower bound for

any realistic algorithm. However, this bound revealed to be rather inaccurate as N grows, due to the strength of the new assumption. This is why we proposed a more general method to derive a bound. By imposing a new assumption, we managed to obtain a new bound, more accurate. However, the computation of this bound requires to solve a system of ODEs, whose size depends on N . As we did not manage to find a general way of solving that system, this method has a major drawback when trying to compute a bound, in the sense that it requires to solve the system for each value of N . Finally, we derived interesting bounds for this problem, and designed a method that can be used to try other assumptions, and then to derive different bounds.

In Chapter 2, we tackled the second problem: the Mean Consensus Problem, in which agents try to estimate the mean of the distribution defining their intrinsic value. In opposition with the previous problem, the objective of the consensus doesn't change with the agents in the system. Hence, we were able to develop the expression of the error induced by the estimation made by an agent, and to properly define a bound based on the amount of knowledge agents rely on to make their estimation. The complexity of the computations did not allow us yet to compute the exact bound. This is why we proceeded to simulations in order to characterize the bound, and what it relies on in a transient state. Thanks to those, we were able to develop a first insight on the expected behavior of the bound, and to build some bases of research that could be a starting point for a more precise elaboration of the desired bound in the future.

At the end of the day, there are many things left to be done in the young research field that represents the study of Open Multi-Agent Systems. In the incoming years, we can expect to see an expansion of the analyses concerning this topic, in the view of the quantity of applications relying on it. Being able to build the bases of such a field will certainly allow huge improvements for the research, but also in many more real-life situations, such as the development of Autonomous vehicles interacting with each other, and many more. This work has been a little part of this study in the definition of fundamental performance limitations. As we said, many things remain to be explored or developed. Some things would deserve a deeper investigation, such as the generalization of the establishment of the bound of the Average Consensus Problem presented in Section 1.4.3, or the formal definition of a proper bound for the Mean Consensus Problem, which has only been approached through simulations for the time being.

Appendices

Appendix A

Local error for the Ping model with three agents

Equivalently to the 2 agents case, we can list all possibilities during the time lapse Δt . By denoting R_i (resp. N_i) the event of having only agent i replaced (resp. not replaced), and R_{-i} (resp. N_{-i}) as the event of having all agents, except agent i replaced (resp. not replaced), we can define the estimation of $\bar{x}$ made by agent 1, and the error induced for all cases combinations.

Using the notations $p_N := e^{-\lambda_r \Delta t}$ and $x_i := x_i(t - \Delta t)$, we have

- $\mathbb{E}[\bar{x}(t)|R_1] = \frac{x_1}{3}$
- $\mathbb{E}[\bar{x}(t)|N_1] = \frac{x_1 + e(x_2 + x_3)}{3}$

Hence,

- $\mathbb{E}[(\bar{x} - y_1)^2|R_1] = \mathbb{E}\left[\frac{x_1}{3} - \frac{x_1 + X_2 + X_3}{3}\right]^2 = \frac{2}{9}\sigma^2$
- $\mathbb{E}[(\bar{x} - y_1)^2|N_1 \cap R_2] = \mathbb{E}\left[\frac{x_1 + p_N(x_2 + x_3)}{3} - \frac{x_1 + X_2 + X_3}{3}\right]^2$
 $= \frac{1}{9}\left(\mathbb{E}[p_N x_2 - X_2]^2 + 2\mathbb{E}[p_N x_2 - X_2]\mathbb{E}[(p_N - 1)x_3] + \mathbb{E}[(p_N - 1)x_3]^2\right)$
 $= \frac{1}{9}[(p_N^2 + 1)\sigma^2 + (p_N - 1)^2\sigma^2] = \frac{2}{9}(p_N^2 - p_N + 1)\sigma^2$
- $\mathbb{E}[(\bar{x} - y_1)^2|N_1 \cap R_3] = \frac{2}{9}(p_N^2 - p_N + 1)\sigma^2$ (by symmetry with above)
- $\mathbb{E}[(\bar{x} - y_1)^2|N_1 \cap R_{-1}] = \mathbb{E}\left[\frac{x_1 + x_2 + x_3}{3} - \frac{x_1 + X_2 + X_3}{3}\right]^2$
 $= \frac{1}{9}\left(\mathbb{E}[p_N x_2 - X_2]^2 - 2\mathbb{E}[p_N x_2 - X_2]\mathbb{E}[p_N x_3 - X_3] + \mathbb{E}[p_N x_3 - X_3]^2\right)$
 $\downarrow$ (Since $\mathbb{E}[p_N x_i - X_i] = 0$)
 $= \frac{1}{9}[(p_N^2 + 1)\sigma^2 + (p_N^2 + 1)\sigma^2] = \frac{2}{9}(p_N^2 + 1)\sigma^2$
- $\mathbb{E}[(\bar{x} - y_1)^2|N_1 \cap N_{-1}] = \mathbb{E}\left[\frac{x_1 + p_N(x_2 + x_3)}{3} - \frac{x_1 + x_2 + x_3}{3}\right]^2$
 $= \mathbb{E}\left[\frac{(p_N - 1)(x_2 + x_3)}{3}\right]^2 = \frac{2}{9}(p_N - 1)^2\sigma^2$

Putting everything together:

$$\begin{aligned}\mathbb{E}[(\bar{x} - y_1)^2|\Delta t] &= \frac{2}{9}\left[(1 - p_N) + p_N(2p_N(1 - p_N)(p_N^2 - p_N + 1) + (1 - p_N)^2(p_N^2 + 1) + p_N^2(p_N - 1)^2)\right]\sigma^2 \\ &= \frac{2}{9}[1 - p_N + p_N(1 - p_N^2)]\sigma^2 = \frac{2}{9}(1 - p_N^3)\sigma^2\end{aligned}$$

Appendix B

Local error for the Ping model with N agents

Let us generalize the result of the local error when the last communication occurred some time Δt ago, obtained for both two and three agents, to the general case of N agents. To do so, keep in mind that the intrinsic values of agents are i.i.d. and uncorrelated. Hence, One can write for any $i \neq j$, if we denote by x_i the value of agent i at some time t , and by $\hat{x}_i$ the estimation of the value of that agent based on an observation done Δt ago

$$\mathbb{E}[x_i - \hat{x}_i] \mathbb{E}[x_j - \hat{x}_j] = (\mathbb{E}[x_i] - e^{-\lambda_r \Delta t} \mathbb{E}[x_i]) \mathbb{E}[x_j - \hat{x}_j] = 0 \cdot \mathbb{E}[x_j - \hat{x}_j] = 0$$

Then, we can write

$$\begin{aligned} \mathbb{E}[(\bar{x} - y_1)^2 | \Delta t] &= \mathbb{E} \left[\left(\frac{\sum_{i=1}^N x_i - \hat{x}_i}{N} \right)^2 \right] = \frac{1}{N^2} \mathbb{E} \left[\sum_{i=1}^N (x_i - \hat{x}_i)^2 + \sum_{i \neq j} (x_i - \hat{x}_i)(x_j - \hat{x}_j) \right] \\ &= \sum_{i=1}^N \frac{\mathbb{E}[x_i - \hat{x}_i]^2}{N^2} + \frac{1}{N^2} \sum_{i \neq j} \mathbb{E}[(x_i - \hat{x}_i)(x_j - \hat{x}_j)] \\ &\quad \downarrow \text{(Since } x_i \text{ and } x_j \text{ are uncorrelated: } \mathbb{E}[x_i x_j] = \mathbb{E}[x_i] \mathbb{E}[x_j]) \\ &= \frac{1}{N^2} \sum_{i=1}^N \mathbb{E}[x_i - \hat{x}_i]^2 + \frac{1}{N^2} \sum_{i \neq j} \mathbb{E}[x_i - \hat{x}_i] \mathbb{E}[x_j - \hat{x}_j] \\ &\quad \downarrow \text{Since } \mathbb{E}[x_j - \hat{x}_j] = 0, \text{ and by symmetry of the agents} \\ &= \frac{N-1}{N^2} \mathbb{E}[x_i - \hat{x}_i]^2 \end{aligned}$$

With this formulation, the expectation to be calculated is actually the error obtained on the estimation of one single agent. This is easily obtained by a similar way we did in Section 1.3.2 for two agents by denoting the same three cases, when the focus is on agent 1 compared to some agent j .

- $\mathbb{E}[(x_j(t) - \hat{x}_j(t))^2 | R_1] = \mathbb{E}[x_j(t)]^2 = \sigma^2$
- $\mathbb{E}[(x_j(t) - \hat{x}_j(t))^2 | N_1 \cap R_j] = \mathbb{E}[x_j(t) - x_j(t-T)p_N]^2 = (1 + p_N^2)\sigma^2$
- $\mathbb{E}[(x_j(t) - \hat{x}_j(t))^2 | N_1 \cap N_j] = \mathbb{E}[(1 - p_N)x_j(t-T)]^2 = (1 - p_N)^2\sigma^2$

Hence, summing up the cases, we have

$$\begin{aligned}\mathbb{E} \left[(x_j - \hat{x}_j)^2 | \Delta t \right] &= (1 - p_N) \mathbb{E} \left[(x_j - \hat{x}_j)^2 | R_i \right] \\ &\quad + p_N \left((1 - p_N) \mathbb{E} \left[(x_j - \hat{x}_j)^2 | N_i \cap R_j \right] + p_N \mathbb{E} \left[(x_j - \hat{x}_j)^2 | N_i \cap N_j \right] \right) \\ &= (1 - p_N^3) \sigma^2\end{aligned}$$

and then

$$\mathbb{E} \left[(\bar{x} - y_1)^2 | \Delta t \right] = \frac{N-1}{N^2} \mathbb{E} \left[(x_j - \hat{x}_j)^2 | \Delta t \right] = \frac{N-1}{N^2} (1 - p_N^3) \sigma^2$$

Appendix C

Bounds for several CDFs: Ping, Ping(t/2) and 1

It is shown in Section 1.4.3 that the individual error on the estimation of an agent j which was last updated T ago is given by $E_j(T) = (1 - e^{-3\lambda_r T})\sigma^2$. From it, we know the error is given by

$$\mathbb{E}[\bar{x} - y_i]^2 = \frac{N-1}{N^2} \int_0^\infty f_C(T) E_j(T) dT$$

We will compute this error for the three choices of CDFs used to depict Proposition 1 in Section 1.4.2.

- Ping

With the Ping method, the CDF is given by $P(t) = 1 - e^{-(N-1)\lambda_c t}$, and thus

$$f_C(t) = (N-1)\lambda_c e^{-(N-1)\lambda_c t}$$

Hence

$$\begin{aligned} \mathbb{E}[\bar{x} - y_i]^2 &= \frac{N-1}{N^2} \int_0^\infty (N-1)\lambda_c e^{-(N-1)\lambda_c t} (1 - e^{-3\lambda_r t}) \sigma^2 dt \\ &= \frac{(N-1)^2}{N^2} \lambda_c \sigma^2 \left[\frac{1}{(N-1)\lambda_c} - \frac{1}{(N-1)\lambda_c + 3\lambda_r} \right] \\ &= \frac{N-1}{N^2} \sigma^2 \left(1 - \frac{3\lambda_r}{(N-1)\lambda_c + 3\lambda_r} \right) = \frac{N-1}{N^2} \left(\frac{1}{1 + \frac{N-1}{3} \frac{\lambda_c}{\lambda_r}} \right) \sigma^2 \end{aligned}$$

This bound is equivalent to the one derived with the Ping model.

- Ping($t/2$)

The interpretation of this CDF is given in Section 1.4.2, and is defined by $P(t) = 1 - e^{-(N-1)\lambda_c t/2}$ which gives

$$f_C(t) = \frac{N-1}{2} \lambda_c e^{-(N-1)\lambda_c t/2}$$

Hence

$$\begin{aligned} \mathbb{E} [\bar{x} - y_i]^2 &= \frac{N-1}{2N^2} \int_0^\infty (N-1) \lambda_c e^{-(N-1)\lambda_c t/2} \left((1 - e^{-3\lambda_r t}) \sigma^2 \right) dt \\ &= \frac{(N-1)^2}{2N^2} \lambda_c \sigma^2 \left[\frac{2}{(N-1)\lambda_c} - \frac{2}{(N-1)\lambda_c + 6\lambda_r} \right] \\ &= \frac{N-1}{N^2} \sigma^2 \left(1 - \frac{6\lambda_r}{(N-1)\lambda_c + 6\lambda_r} \right) = \frac{N-1}{N^2} \left(\frac{1}{1 + \frac{N-1}{6} \frac{\lambda_c}{\lambda_r}} \right) \sigma^2 \end{aligned}$$

- Full knowledge: $P(t) = 1$

In this situation, we assume agents always know everything: $P(t) = 1 \Rightarrow f_C(t) = 0$.

Hence,

$$\mathbb{E} [\bar{x} - y_i]^2 = \frac{N-1}{N^2} \int_0^\infty 0 \cdot E_j(T) dT = 0$$

Appendix D

Computation of the Infection bound by hand

In this part, we will develop the resolution by hand of the systems of ODEs defined in Section 1.4.3 for 2, 3 and 4 agents. The definition of the system is such that each ODE to find $P_k(t)$ only depends on $P_k(t)$ and the previous result $P_{k-1}(t)$ (with $P_0(t) = 0 \forall t \geq 0$). Hence, we can proceed by cascading resolution, which is what is done here. As a reminder

$$\dot{P}_k(t) = \lambda_c(k-1)(N-k+1)P_{k-1}(t) - \lambda_c k(N-k)P_k(t)$$

with $P_k(0) = 0 \forall k > 1$ and $P_1(0) = 1$.

- $N = 2$

1. $\dot{P}_1(t) = -\lambda_c P_1(t) \Rightarrow P_1(t) = P_1(0)e^{-\lambda_c t} = e^{-\lambda_c t}$
2. $\dot{P}_2(t) = (k-1)\lambda_c P_1(t) = \lambda_c e^{-\lambda_c t} \Rightarrow P_2(t) = P_2(0) + \int_0^t \lambda_c e^{-\lambda_c t^*} dt^* = 1 - e^{-\lambda_c t}$

Hence

$$P(t) = \frac{0}{1}P_1(t) + \frac{1}{1}P_2(t) = 1 - e^{-\lambda_c t}$$

This is the same CDF as the Ping model for two agents, and leads thus to the same error (see Appendix C for details):

$$\mathbb{E}[\bar{x} - y_i]^2 = \frac{1}{4} \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r}} \right) \sigma^2$$

- $N = 3$

1. $\dot{P}_1(t) = -2\lambda_c P_1(t) \Rightarrow P_1(t) = P_1(0)e^{-2\lambda_c t} = e^{-2\lambda_c t}$
2. $\dot{P}_2(t) = 2\lambda_c P_1(t) - 2\lambda_c P_2(t) \Rightarrow P_2(t) = P_2^h(t) + P_2^p(t)$
Homogeneous : $P_2^h(t)$

$$\dot{P}_2^h(t) = -2\lambda_c P_2^h(t) \Rightarrow P_2^h(t) = K e^{-2\lambda_c t}$$

Particular : $P_2^p(t)$: Variation of parameters method with $K = K(t)$

$$\begin{aligned} \dot{P}_2(t) &= \dot{K} e^{-2\lambda_c t} - 2\lambda_c K e^{-2\lambda_c t} = 2\lambda_c e^{-2\lambda_c t} - 2\lambda_c K e^{-2\lambda_c t} \Rightarrow K(t) = 2\lambda_c t \\ &\Rightarrow P_2^p(t) = 2\lambda_c t e^{-2\lambda_c t} \end{aligned}$$

Then $P_2(t) = K e^{-2\lambda_c t} + 2\lambda_c t e^{-2\lambda_c t}$ and

$$P_2(0) = K = 0 \Rightarrow P_2(t) = 2\lambda_c t e^{-2\lambda_c t}$$

3. $\dot{P}_3(t) = 2\lambda_c P_2(t) = 4\lambda_c^2 t e^{-2\lambda_c t}$, thus

$$\begin{aligned}
P_3(t) &= P_3(0) + \int_0^t 4\lambda_c^2 t^* e^{-2\lambda_c t^*} dt^* \\
&= \left[\frac{4\lambda_c^2 t^* e^{-2\lambda_c t^*}}{-2\lambda_c} \right]_0^t - \int_0^t \frac{4\lambda_c^2 e^{-2\lambda_c t^*}}{-2\lambda_c} dt^* \\
&= -2\lambda_c t e^{-2\lambda_c t} - \left[\frac{4\lambda_c^2 e^{-2\lambda_c t^*}}{4\lambda_c^2} \right]_0^t = 1 - 2\lambda_c t e^{-2\lambda_c t} - e^{-2\lambda_c t} \\
&= 1 - (1 + 2\lambda_c t) e^{-2\lambda_c t}
\end{aligned}$$

Hence

$$P(t) = \frac{0}{2}P_1(t) + \frac{1}{2}P_2(t) + \frac{2}{2}P_3(t) = 1 - (1 + \lambda_c t) e^{-2\lambda_c t}$$

We derive from $P(t)$ the pdf

$$f_j(t) = \lambda_c(2\lambda_c + 1 - e^{-2\lambda_c t})$$

Hence, integrating, we obtain the bound for three agents from the error of estimation of an agent

$$\begin{aligned}
\mathbb{E}[x_j - \hat{x}_j]^2 &= \int_0^\infty \lambda_c(2\lambda_c + 1) e^{-2\lambda_c t} (1 - e^{-3\lambda_r t}) \sigma^2 dt \\
&= \lambda_c \sigma^2 \int_0^\infty 2\lambda_c t e^{-2\lambda_c t} - 2\lambda_c t e^{-(2\lambda_c + 3\lambda_r)t} + e^{-2\lambda_c t} - e^{-(2\lambda_c + 3\lambda_r)t} dt \\
&= \lambda_c \sigma^2 \left(0 - \frac{2\lambda_c}{4\lambda_c^2} - 0 + \frac{2\lambda_c}{-(2\lambda_c + 3\lambda_r)^2} + \frac{1}{-2\lambda_c} - \frac{1}{-(2\lambda_c + 3\lambda_r)} \right) \\
&= \lambda_c \sigma^2 \left(\frac{1}{\lambda_c} - \frac{2\lambda_c}{(2\lambda_c + 3\lambda_r)^2} - \frac{1}{2\lambda_c + 3\lambda_r} \right) = \sigma^2 \left(1 - \frac{4\lambda_c^2 + 3\lambda_c \lambda_r}{(2\lambda_c + 3\lambda_r)^2} \right) \\
&= \sigma^2 \left(\frac{9\lambda_c \lambda_r + 9\lambda_r^2}{4\lambda_c^2 + 12\lambda_c \lambda_r + 9\lambda_r^2} \right) = \sigma^2 \left(\frac{1}{1 + \frac{4\lambda_c^2 + 3\lambda_c \lambda_r}{9\lambda_c \lambda_r + 9\lambda_r^2}} \right)
\end{aligned}$$

Finally, with a little more algebra, we obtain

$$\mathbb{E}[\bar{x} - y_i]^2 = \frac{2}{9} \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r} \left(1 + \frac{\lambda_c}{3(\lambda_c + \lambda_r)} \right)} \right) \sigma^2$$

- $N = 4$

1. $\dot{P}_1(t) = -3\lambda_c P_1(t) \Rightarrow P_1(t) = P_1(0)e^{-3\lambda_c t} = e^{-3\lambda_c t}$
2. $\dot{P}_2(t) = 3\lambda_c P_1(t) - 4\lambda_c P_2(t) \Rightarrow P_2(t) = P_2^h(t) + P_2^p(t)$
Homogeneous : $P_2^h(t)$

$$\dot{P}_2^h(t) = -4\lambda_c P_2^h(t) \Rightarrow P_2^h(t) = K e^{-4\lambda_c t}$$

Particular : $P_2^p(t)$: Variation of parameters method with $K = K(t)$

$$\begin{aligned} \dot{P}_2(t) &= \dot{K} e^{-4\lambda_c t} - 4\lambda_c K e^{-4\lambda_c t} = 3\lambda_c e^{-3\lambda_c t} - 4\lambda_c K e^{-4\lambda_c t} \Rightarrow K(t) = 3e^{\lambda_c t} \\ \Rightarrow P_2^p(t) &= 3e^{\lambda_c t} e^{-4\lambda_c t} = 3e^{-3\lambda_c t} \end{aligned}$$

Then $P_2(t) = K e^{-4\lambda_c t} + 3e^{-3\lambda_c t}$ and

$$P_2(0) = 3 + K = 0 \Rightarrow P_2(t) = 3e^{-3\lambda_c t} - 3e^{-4\lambda_c t}$$

3. $\dot{P}_3(t) = 4\lambda_c P_2(t) - 3\lambda_c P_3(t) \Rightarrow P_3(t) = P_3^h(t) + P_3^p(t)$
Homogeneous : $P_3^h(t)$

$$\dot{P}_3^h(t) = -3\lambda_c P_3^h(t) \Rightarrow P_3^h(t) = K e^{-3\lambda_c t}$$

Particular : $P_3^p(t)$: Variation of parameters method with $K = K(t)$

$$\begin{aligned} \dot{P}_3(t) &= \dot{K} e^{-3\lambda_c t} - 3\lambda_c K e^{-3\lambda_c t} = 12\lambda_c e^{-3\lambda_c t} - 12\lambda_c e^{-4\lambda_c t} - 3\lambda_c K e^{-3\lambda_c t} \\ \Rightarrow K(t) &= 12\lambda_c t + 12e^{-\lambda_c t} \\ \Rightarrow P_3^p(t) &= (12\lambda_c t + 12e^{-\lambda_c t}) e^{-3\lambda_c t} \end{aligned}$$

Then $P_3(t) = K e^{-3\lambda_c t} + (12\lambda_c t + 12e^{-\lambda_c t}) e^{-3\lambda_c t}$ and

$$P_3(0) = K + 12 = 0 \Rightarrow P_3(t) = 12(\lambda_c t - 1)e^{-3\lambda_c t} + 12e^{-4\lambda_c t}$$

4. $\dot{P}_4(t) = 3\lambda_c P_3(t) = 36\lambda_c (\lambda_c t e^{-3\lambda_c t} - e^{-3\lambda_c t} + e^{-4\lambda_c t})$, thus

$$\begin{aligned} P_4(t) &= P_4(0) + \int_0^t 36\lambda_c^2 t^* e^{-3\lambda_c t^*} - 36\lambda_c e^{-3\lambda_c t^*} + 36\lambda_c e^{-3\lambda_c t^*} dt^* \\ &= \left[\frac{36\lambda_c^2 t^* e^{-3\lambda_c t^*}}{-3\lambda_c} \right]_0^t - \int_0^t \frac{36\lambda_c^2 e^{-3\lambda_c t^*}}{-3\lambda_c} dt^* - \left[\frac{36\lambda_c e^{-3\lambda_c t^*}}{-3\lambda_c} \right]_0^t + \left[\frac{36\lambda_c e^{-4\lambda_c t^*}}{-4\lambda_c} \right]_0^t \\ &= -12\lambda_c t e^{-3\lambda_c t} - \left[\frac{36\lambda_c^2 e^{-3\lambda_c t^*}}{9\lambda_c^2} \right]_0^t - (12 - 12e^{-3\lambda_c t}) + (9 - 9e^{-4\lambda_c t}) \\ &= -12\lambda_c t e^{-3\lambda_c t} + 4 - 4e^{-3\lambda_c t} - 12 + 12e^{-3\lambda_c t} + 9 - 9e^{-4\lambda_c t} \\ &= 1 + (8 - 12\lambda_c t) e^{-3\lambda_c t} - 9e^{-4\lambda_c t} \end{aligned}$$

Hence

$$P(t) = \frac{0}{3}P_1(t) + \frac{1}{3}P_2(t) + \frac{2}{3}P_3(t) + \frac{3}{3}P_4(t) = 1 + (1 - 4\lambda_c t)e^{-3\lambda_c t} - 2e^{-4\lambda_c t}$$

We derive from $P(t)$ the pdf

$$f_j(t) = \lambda_c \left(-7 + 12\lambda_c t + 8e^{-\lambda_c t} \right) e^{-3\lambda_c t}$$

Hence, integrating, we obtain the bound for four agents from the error of estimation of an agent

$$\begin{aligned} \mathbb{E}[x_j - \hat{x}_j]^2 &= \int_0^\infty \lambda_c \left(-7 + 12\lambda_c t + 8e^{-\lambda_c t} \right) e^{-3\lambda_c t} \left(1 - e^{-3\lambda_r t} \right) dt \\ &\quad \downarrow \text{By integrating each of the six terms separately} \\ &= \lambda_c \sigma^2 \left(\frac{2}{\lambda_c} + \frac{4}{3\lambda_c} - \frac{7}{3\lambda_c} - \frac{8}{4\lambda_c + 3\lambda_r} - \frac{12\lambda_c^2}{(3\lambda_c + 3\lambda_r)^2} + \frac{7\lambda_c}{3\lambda_c + 3\lambda_r} \right) \\ &= \sigma^2 \left(\frac{66\lambda_c^2\lambda_r + 81\lambda_c\lambda_r^2 + 27\lambda_r^3}{36\lambda_c^3 + 99\lambda_c^2\lambda_r + 90\lambda_c\lambda_r^2 + 27\lambda_r^3} \right) \\ &= \sigma^2 \left(\frac{1}{1 + \frac{36\lambda_c^3 + 33\lambda_c^2\lambda_r + 9\lambda_c\lambda_r^2}{66\lambda_c^2\lambda_r + 81\lambda_c\lambda_r^2 + 27\lambda_r^3}} \right) \end{aligned}$$

Finally, with a little more algebra, we obtain

$$\mathbb{E}[\bar{x} - y_i]^2 = \frac{3}{16} \left(\frac{1}{1 + \frac{1}{3} \frac{\lambda_c}{\lambda_r} \left(1 + \frac{14\lambda_c^2 + 6\lambda_c\lambda_r}{22\lambda_c^2 + 27\lambda_c\lambda_r + 9\lambda_r^2} \right)} \right) \sigma^2$$

Bibliography

- [1] L.J. LUOTSINEN. *Autonomous Environmental Mapping in Multi-Agent UAV Systems*. Electronic Theses and Dissertations, University of Central Florida, 2002.
- [2] M. SELECKÝ, T. MEISER. *Integration of Autonomous UAVs into Multi-agent Simulation*. Journal of Advanced Engineering, vol. 52, n. 5, 2012.
- [3] C. MASSERA, M. TERRA, D. WOLF. *Safely Optimizing Highway Traffic with Robust Model Predictive Control-based Cooperative Adaptive Cruise Control*. IEEE Transactions on Intelligent Transportation Systems, vol. 18, n. 11, 2017.
- [4] A. PRUSIEWICZ. *A Multi-agent System for Computer Network Security Monitoring*. Agent and Multi-Agent Systems: Technologies and Applications, pp 842-849, 2008.
- [5] A. NEDIĆ, A. OLSHEVSKY, M.G. RABBAT. *Network Topology and Communication-Computation Tradeoffs in Decentralized Optimization*. Proceedings of the IEEE, vol. 106, n. 5, 2018.
- [6] F. YAN, S. SUNDARAM, S.V.N. VISHWANATHAN, Y. QI. *Distributed Autonomous Online Learning: Regrets and Intrinsic Privacy-Preserving Properties*. IEEE Transactions on Knowledge and Data Engineering, vol. 25, n. 11, 2013.
- [7] J. DUCHI, A. AGARWAL, M.J. WAINWRIGHT. *Dual Averaging for Distributed Optimization: Convergence Analysis and Network Scaling (Distributed Dual Averaging in Networks)*. IEEE Transactions on Automatic Control, vol. 57, n. 3, 2011.
- [8] M. ABDELRAHIM, J. HENDRICKX, M. HEEMELS. *MAX-consensus in open multi-agent systems with gossip interactions*. Proceedings of the 56th IEEE Conference on Decision and Control, 2017.
- [9] J. HENDRICKX, S. MARTIN. *Open Multi-Agent Systems: Gossiping with Random Arrivals and Departures*. Proceedings of the 56th IEEE Conference on Decision and Control, 2017.
- [10] A. NEDIĆ, A. OLSHEVSKY. *Distributed Optimization over time-varying directed graphs*. IEEE Transactions on Automatic Control, vol. 60, n. 3, 2015.
- [11] J. NIRJHOR, D. BHAT, K. JAISWAL, AND 5 MORE. *Jensen's inequality*. Brilliant, online, last consulted on 6 june 2018, <https://brilliant.org/wiki/jensens-inequality/>

