UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

BERNARDO HUMMES FLORES

Formalization of Mobile Robot Tasks

Work presented in partial fulfillment of the requirements for the degree of Bachelor in Computer Science

Advisor: Prof. Dr. Mariana Kolberg
Co-advisor: Prof. Dr. Eric Goubault

Porto Alegre
October 2022

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Reitor: Prof. Carlos André Bulhões
Vice-Reitora: Profª. Patricia Pranke
Pró-Reitora de Graduação: Profª. Cíntia Inês Boll
Diretora do Instituto de Informática: Profª. Carla Maria Dal Sasso Freitas
Coordenador do Curso de Ciência de Computação: Prof. Marcelo Walter
Bibliotecário-chefe do Instituto de Informática: Alexsander Borges Ribeiro

ABSTRACT

The guaranteed coordination and operation of mobile robots without supervision requires tools capable of representing the vast number of evolving possibilities during a mission. This work studies the categorical and logical formalization of mobile robot tasks, with the use of connections between robotics, distributed computing and modal logics.

A demonstration of the equivalences between modal logics and combinatorial topology is shown as the first element for the unified perspective, alongside some of its uses. This is followed by a discussion on the connections between models in distributed computing and robotics, where initial results on the formalization of robot tasks are accompanied by considerations on applications for gathering and exploration missions.

The work ends by proposing a formalization that extends the previous one in the literature with the capacity of dealing with faulty robots, limited sensing capabilities and uncertain behavior.

Keywords: Mobile Robotics. Distributed Computing. Formal Verification.

Formalização de Tarefas de Robôs Móveis

RESUMO

A coordenação e operação de robôs sem supervisão requere ferramentas capazes de representar o vasto número de possíveis evoluções de uma missão. Este trabalho é um estudo da formalização categórica e lógica de tarefas de robôs móveis, fazendo uso de conexões entre robótica, computação distribuída e lógica modal.

Uma demonstração da equivalências entre lógica modal e topologia combinatória é exibida como o primeiro elemento de uma perspectiva unificada, junto dos seus usos. Uma discussão segue com as conexões entre modelos em computação distribuída e robótica, onde resultados iniciais na formalização de tarefas de robôs são acompanhados por considerações nas seus usos para missões de encontro e exploração.

Este trabalho encerra com a proposta de uma formaçização que extende a anterior encontrada na literatura com a capacidade de lidar com robôs defeituosos, capacidades limitadas de sensoriamento e comportamento incerto.

Palavras-chave: Robótica Móvel. Computação Distribuída. Verificação Formal.

LIST OF FIGURES

CONTENTS

1 THE CONTEXT

1.1 Introduction

This work is the study of links between computing models for mobile robotics and similar structures in distributed systems and modal logics, with the goal of providing new tools for the analysis of guaranteed results in distributed robotic tasks. It is often hard to reason about the feasibility of scenarios where multiple robots have to cooperate towards a common objective, as there is an explosion of possible executions without a common controller for all agents. Furthermore, the possibility of failures has to be addressed, as well as the resistance of a system to their occurrences. This will be tackled in a proposed formalization of robot tasks, alongside the envisioned expansions and uses, such as with gathering and exploration missions.

Guaranteed results in distributed computing have been long approached with algebraic topology, with many approaches explored in Herlihy, Kozlov and Rajsbaum (2014), where the possible executions and the associated evolution of the knowledge within the agents can be understood as continuous transformations from an initial configuration represented as a topological element. This has allowed for the inference of possibility and impossibility results of an algorithm's capacity to solve a given task accompanied by the system specification. Recent results, such as in Alcántara et al. (2019), in this domain showed the similarity with the theoretical models in robotics, opening the exploration and importation of known results.

In parallel, the similarity with mathematical structures in modal logic can be observed from a more abstract point of view, such as the one from category theory. The aforementioned topological structure is strongly connected to Kripke frames used in modal logics, which allows for the interpretation of an interesting semantics that expands even more the possible importation of results, with some examples in Goubault, Ledent and Rajsbaum (2018), Goubault, Ledent and Rajsbaum (2022). Considerations on this new point of view proposed usage will be shown, as the natural interpretation of knowledge and time can be advantageous concepts to improve the expressiveness of the formalization.

Those connections will be exposed in the following chapters, and the necessary theoretical background can be found in the appendix 2. Considerations on the current research for each of the fields and their interactions will start with modal logic and combinatorial topology in chapter 3, then the distributed computing approach used for robots in chapter 4. A proposed formalization of robot tasks will be presented in chapter 5, alongside its relation to logic. Finally, in chapter 6 the progress of the goals is discussed, followed by what is imagined as future work, such as other ways to profit of this abstract point of view that connects the different fields.

1.2 Related work

It is possible to identify some categories of previous works that were used. The first one consists of the existing problems and their solutions in robotics within the theoretical robotics model of Look-Compute-Move (LCM). Next, we have the existing efforts in usage of modal logic and distributed computing together, with complementing approaches. The same happens with distributed computing and theoretical robotics, again with the LCM model. This is where most of the efforts in this work are found, as it has only recently started being explored. Finally we have the references with the theory behind each of the tackled domains.

We will be considering only cases of robot gathering and exploration, that in discretized environments of graphs/networks, as opposed to models that operate in Euclidean spaces, such as in Luna et al. (2022), Kirkpatrick et al. (2021). The variations that we are the most interested in are the cases with lights for implicit communication where one robot has to see the exposed information, seen in Nakai, Sudo and Wada (2021), Das et al. (2016), Alcántara et al. (2019), and of limited visibility, where there is a maximum range to some or all of the robot's sensors, seen in Flocchini et al. (2005), Datta et al. (2013a), Guerraoui and Maurer (2019). In some cases they are combined, but it is possible to understand better how the transmission of information influences the solvability with myopic robots in the works of Ooshita and Tixeuil (2021), Nagahama, Ooshita and Inoue (2021), Bramas, Lafourcade and Devismes (2021).

The links between distributed computing and logic have been made with different levels of equivalences among the topological structures used in representing distributed executions, simplicial complexes, and the Kripke frames, usually in epistemic logic, presented first in Goubault, Ledent and Rajsbaum (2018) using dynamic epistemic logic. Variations were explored, with additional information represented, via complex and Kripke models, and with varying number of agents, seen in Armenta-Segura, Rajsbaum and Ledent (2020), Ditmarsch et al. (2020), Goubault, Ledent and Rajsbaum (2022). The idea of mixing modalities in order to represent time and knowledge has been explored in Knight (2013), with the aim of the current work being expanded on this formalization.

The main result relating distributed computing and theoretical robotics has been presented in Alcántara et al. (2019), where a model of asynchronous computation in shared memory was shown equivalent to the extended case of asynchronous robots with lights in a graph. This allowed for the first major importation of results between domains. Another interesting result in abstraction was made in Ledent (2021), where multiple cases of gathering were shown to be variations of a more generic problem, allowing for stronger implications on their solvability. These key results will be expanded later.

As mentioned, a brief introduction of the necessary concepts will be presented in the following chapter, but a reader may find further explanations beyond the scope of this work in the following texts. The abstract representation of robotic operations known as Look-Compute-Move is presented with multiple variations in the collective of publications Flocchini, Prencipe and Santoro (2019), and the chapters on gathering and exploration are useful surveys of the area. Together with the book Herlihy, Kozlov and Rajsbaum (2014), it is possible to observe the theoretical base and for most of what is dealt in this work and the previously mentioned references. The work Fajstrup et al. (2016) can also be useful for those interesting in learning more about the mentioned usage of topology in concurrent applications.

In the case of modal logics Benthem (2010) provides interesting insights on the power of this formalism and on the usage of Kripke frames. A more extended work on the temporal and epistemic systems in use can be found in Carnielli and Pizzi (2008). We used the language of category theory, while Lane (1971) is the original work on it, Riehl (2016) presents more contextualized explanations, for those coming from different fields in mathematics, Barr and Wells (1990) does so with a focus in computer scientists' background and Smith (2018) offers a slower pace in developing the intuition.

2 THEORETICAL BACKGROUND

2.1 Distributed computing

Here, some notes expose the base for the study of distributed computation via topological properties, mainly from Herlihy, Kozlov and Rajsbaum (2014).

2.1.1 Introduction

The usage of topology in order to represent the evolving knowledge of multiple agents makes use of an interpretation of simplicial complexes, where the multiple possible states are connected according to the associated possible scenarios. The following cases will demonstrate how this analysis is made in simple cases and motivate for the more complex constructions ahead.

2.1.1.1 The muddy children problem

This initial problem instantiates how knowledge evolves via communication of agents that begin without knowing their own states. One possible enunciation is as follows:

> A group of children is playing in the garden, and some of them end up with mud on their foreheads. Each child can see the other children's foreheads but not his or her own. At noon, their teacher summons the children and says: "At least one of you has a muddy forehead. You are not allowed to communicate with one another about it in any manner. But whenever you become certain that you are dirty, you must announce it to everybody, exactly on the hour." The children resume playing normally, and nobody mentions the state of anyone's forehead. There are six muddy children, and at 6:00 they all announce themselves. How does this work?

This is an idealized problem that shows how information can be gained from each round and that the announcement of existing at least one children with mud in their forehead allows this to happen. If we reduce this problem to only 3 children, it get easier to see how the problem is solved if at least a single one is muddy in the following steps:

1. The professor announces that there is at least one muddy children, removing the option of everyone being clean.
2. Everyone knows each other's foreheads, which means that each person can either:
    1. See no other muddy forehead, which means that they are the muddy one and

    announce in the next communication round;
   2. See one or two other muddy foreheads, which does not give enough information about themselves.

3. Having seen one or two muddy foreheads they can:
   1. If they saw only one other muddy, they know they must be the second and announce in the next round;
   2. If they saw two, they don't have enough information and wait.

3. Having seen two children with mud and two rounds with no announcements, they all announce that they have a muddy forehead.

This is equivalently represented in the following simplicial complex, where each solid triangle has a possible configuration of the three children, each with one color, and considering that they don't know their own state. This representation consists of each vertex representing a possible state of the children, muddy or not. Each state is connected with edges to all others that could be true (it is indistinguishable) while they are true. The following image shows all possible scenarios.

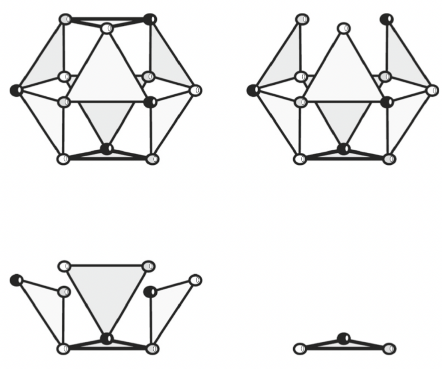


Figure 2.1 – Evolution of the complex through the rounds of (lack of) communication, credit from Herlihy, Kozlov and Rajsbaum (2014).

The triangle on top represents all three children are clean, while each individual vertex of this triangle represents each children being clean, with the different colors distinguishing them. The facets represent global states (in this case triangles), in the form 111 or $01\bot$, where 0 and 1 represent muddy or not, and $\bot$ the lack of information This shows how each vertex has meaning even without context. From this one must note that each vertex connects exactly two triangles, which are the two possible scenarios where this children is the only one that changes between muddy or not. These connections, when interpreted as possible worlds, show that for children A with no knowledge about itself, the world where children B sees no mud in the other ones, children C sees no mud on the other ones, children B sees mud in A but not in C, and children C sees mud in A but not in B are equivalent.

As the announcement in step 1 happens, the top triangle is removed and the top vertices are "exposed", left without ambiguity. This means that, in this case, a children will have enough information to announce it, as in step 2.a. This process repeats in the case of no announcements, leaving the possibilities of $\bot 01, 0\bot 1, 01\bot, \bot 10, 1\bot 0, 10\bot$. If some of this scenarios is fulfilled, then there will be no more lack of information and they announce, as in step 3.1; In step 4 we see them simply having the option of being all muddy.

2.1.1.2 Two generals problem

This second problem shows how in some cases failures in communication is enough for making it impossible to reach a decision. The enunciation of the problem is as follows.

> Two army division, one commanded by general ALice and one by general Bob, are camped on two hilltops overlooking a valley, The enemy is camped in the valley. If both divisions attack simultaneously, they will win, but if only one division attacks by itself, it will be defeated. As a result, neither general will attack without a guarantee that the other will attack at the same time. In particular, **neither general will attack without communication from the other**.
>
> At the time the divisions are deployed on hilltops, **the generals had not agreed on whether or when to attack**. Now Alice decides to schedule an attack. **The generals can communicate only by messengers**. Normally it takes a messenger exactly one hour to get from one encampment to the other. However, it is possible that he will get lost in the dark or, worse yet, be capture by the enemy. Fortunately, **on this particular night all the messenger happen to arrive safely**. How long will it take Alice and Bob to coordinate their attack?

We are interested in arriving in a common decision for both agents, so that either the attack and the decision to stay put are shared. However, one can see that no number of messages will be enough for this to be agreed upon, as the resulting scenario after a message is received or lost is the same for those who sent it. Another confirmation will be required and this process will never end. The delay is not considered here, as this problem is impossible even without taking this into consideration. It is possible to see in image _ the possible states.

At first, we can see that Bob, without any messages received, consider equally the possibility of Alice having sent either the command to attack at dawn or to attack at noon. Once a message is received, Alice will see as equally possible the case where Bob received the message or it has not. This same pattern will repeat forever, and we can

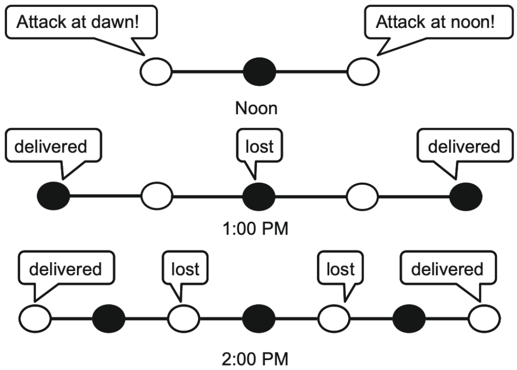


Figure 2.2 – The evolving states after each general sends a message, credit to Herlihy, Kozlov and Rajsbaum (2014).

say that topologically it will always have one connected component. The impossibility appears when we compare with what are the desired configurations, with three different scenarios where they either both attack at dawn, at noon or not at all, seen in figure 2.3. Topologically this constitutes three disconnected components, which is impossible to map functionally from the previous space. This will be further explained in sections 2.1.3 and 2.1.4, where the precise requirements for solvability are exposed.

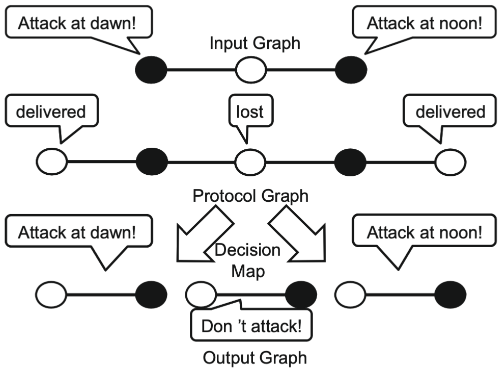


Figure 2.3 – The impossible mapping for the generals communication, credit to Herlihy, Kozlov and Rajsbaum (2014).

2.1.2 Simplicial complexes

2.1.2.1 Definitions

The representation via graphs is limited to at most one failing process, allowing for more on the analysis of specific cases such as consensus with synchronous communication. In order to study general tasks with models where more than once process can fail, *simplicial complexes* are useful, as higher order graphs.

Definition 2.1 *A* ***simplicial complex*** *in the combinatorial view is a pair $\mathcal{K} = (\mathcal{V}, \mathcal{S})$, where V is a finite set of vertices and $S \subseteq \mathcal{P}(V)$ is a set of simplices, non-empty subsets of V, where:*

- *For every $v \in V, \{v\} \in S$. The sets of singleton vertices are part of the set of simplices*
- *S is downward-closed, i.e. for every $Y \in S$ and for every non-empty subset $X \subseteq Y, X \in S$. All of the non-empty subsets of a simplex are part of the set of simplices too, and we say that X is a face of Y.*

The dimension of a *simplex* X is $dim(X) = card(X) - 1$, where the cardinality is the number of edges, and the $codim(\sigma, A) = dim(A) - dim(\sigma)$, where facets have codimension 0. The dimension of a *simplicial complex* $\mathcal{K}$ will be the maximal dimension of a simplex of $\mathcal{K}$, i.e. $dim(\mathcal{K}) = \max\{\dim(\mathcal{X}) | \mathcal{X} \in \mathcal{S}\}$. Vertices $v \in V$ are usually identified as a 0-dimensional simplex $\{v\} \in S$. The maximal simplexes are *facets*, where their definition of being strictly contained $\Delta \subset C$ is only respected w.r.t. the "original" complex. A simplicial complex is *pure* of dimension n if all facets are of the same dimension n. Graphs can be seen as pure 1 dimensional simplicial complexes, with maximal simplexes of dimension 1 (i.e. edges), "glued" together. We can see an example in figure 2.4. Faces can be obtained from excluding a subset of vertices, such as the triangle defined over $\{1, 2, 3\}$ has as its third face $\{1, 2, \hat{3}\}$, seen in figure 2.5.

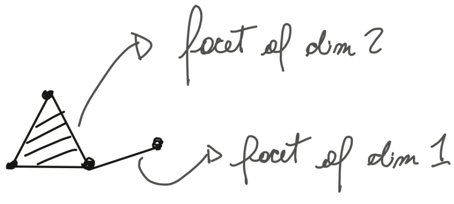


Figure 2.4 – Non pure simplicial complex of dimension 2 (maximal simplex of dimension 2, i.e. triangle).

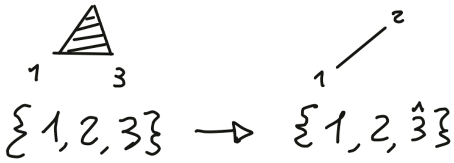


Figure 2.5 – 3rd face of a triangle.

We can define new complexes from an existing one, such as the l-skeleton of C, also written $skel^l(C)$, with the simplicies of C with dimension at most l, e.g. $skel^0(C)$ is the set of vertices. 2^σ is the complex containing σ and all of its faces, and usually is the same as σ. $\partial 2^\sigma$ (or $\partial\sigma$) is the boundary complex of σ, the set of proper faces (facets), and can also be defined as $skel^{n-1}\sigma$, where n is the dimension of σ.

It is also possible to give a geometric explanation for simplicial complexes, into its geometric realization $|\mathcal{A}|$, where we have simplexes "glued" by the edges. They live in $\mathbb{R}^d$ and the focus is on the occupied a space and related topology. It is possible to generate one abstract simplicial complex from a geometric simplicial complex, but many in the other direction.

2.1.2.2 Coloring

Together with the idea of processes in a distributed system, it becomes relevant to identify them in the structures used. This is done via the coloring of vertices, where given a set A of agents, we have:

Definition 2.2 *A chromatic simplicial complex is a pair $\langle K, \mathcal{X}\rangle$, consisting of a simplicial complex $\mathcal{K}$ and a coloring map $\mathcal{X} : \mathcal{V}(K) \to A$, such that for all simplexes $X \in \mathcal{K}$, all vertices of X have distinct colors.*

Each color is used to represent a different process/agent/robot in the system. Note that a coloring consists of a labeling, but not the converse, as not all labeling functions respect the fact that vertices sharing an edge should not have the same color.

2.1.2.3 Maps

In order to express the communication and decision of the processes, the mapping of simplicial complexes via *carrier maps* and *simplicial maps* is used.

Definition 2.3 *A **simplicial map** is a morphism between two simplicial complexes μ : $\mathcal{A} \to \mathcal{B}$, where $\mathcal{V}(\mathcal{A})$ is mapped to $\mathcal{V}(\mathcal{B})$, such that if $\{s_0, s_1, ..., s_n\}$ is a simplex in $\mathcal{A}$, then $\{\mu(s_0), \mu(s_1), ..., u(s_n)\}$ will be a simplex in $\mathcal{B}$. Vertices are sent to vertices, and simplexes to simplexes, where the mapping of the vertices induces the simplexes.*

The mapping of distinct vertices does not have to lead to distinct vertices, but when that happens the simplicial map is called *rigid*, and the image has the same dimension, i.e. $|\sigma| = |\psi(\sigma)|$. A chromatic simplicial map will always be rigid, and equivalent to a *graph homomorphism* in dimension 1, where there is no merger of vertices. Simplicial maps are approximations of continuous maps in the same way that simplicial complexes approximate topological spaces.

Definition 2.4 *Let $\mathcal{A}$ and $\mathcal{B}$ be two simplicial complexes, a* ***carrier map*** *is a function $\Phi : \mathcal{A} \to \mathcal{B}$, where each simplex $\sigma \in \mathcal{A}$ is mapped into a subcomplex of $\mathcal{B}$, such that monotonicity is respected. That is, for all $\sigma, \tau \in \mathcal{A}$, if $\sigma \subseteq \tau$, then $\Phi(\sigma) \subseteq \Phi(\tau)$. This can also be expressed as $\Phi(\sigma \cap \tau) \subseteq \Phi(\sigma) \cap \Phi(\tau)$, meaning that the intersection of the mapping may add information, but will never lose.*

The carrier map Φ is called *strict* if no information is added, that it, $\Phi(\sigma \cap \tau) = \Phi(\sigma) \cap \Phi(\tau)$, in which case, each simplex $\tau \in \Phi(\mathcal{A})$ has a unique carrier in $\mathcal{A}$, that is, a simplex σ of smallest dimension where $\tau \in \Phi(\sigma)$. It is considered *rigid* if every simplex $\sigma \in \mathcal{A}$ and its image $\Phi(\sigma)$ are of the same dimension d. A *chromatic carrier map* will be *rigid* and preserving colors, i.e. for every simplex $\sigma \in \mathcal{A}$, we have that $\mathcal{X}(\sigma) = \mathcal{X}(\Phi(\sigma))$. The image of $\Phi(\mathcal{A})$ will be the union of all subcomplexes of $\mathcal{B}$ that are accessed via $\Phi(\sigma)$, with all $\sigma \in \mathcal{A}$.

The carrier of τ (or $Car(\tau, \Phi(\sigma))$) is the unique simplex $\sigma \in \mathcal{A}$ of smallest dimension, such that $\tau \in \Phi(\sigma)$, for each simplex $\tau \in \Phi(\mathcal{A})$. σ is the unique smallest simplex of $\mathcal{A}$ that maps to the simplex τ of the subcomplex obtained from $\Phi(\mathcal{A})$. Each τ has its carrier.

2.1.2.4 Constructions

We can construct different simplicial complexes and topological spaces from a given complex.

The *star complex* $Star(\sigma, \mathcal{K})$ is defined from another complex $\mathcal{K}$ and a simplex σ. It corresponds to all facets of $\mathcal{K}$ that contain σ. Similarly, a topological space open in the complex is defined with the *open star* $Star^O(\sigma, \mathcal{K})$, composed of the interiors of the simplexes containing σ. We can define an open covering of $|\mathcal{K}|$ with $Star^O(v)_{v \in V(\mathcal{K})}$, where all vertices of $\mathcal{K}$ are considered. If nothing but a simplex σ is specified, the resulting topology will be the intersection of all open stars in its vertices, i.e. $St^O(\sigma) = \bigcap_{v \in V(\sigma)} St^O(v)$.

From the star operator, we can also define a a *link complex* from $Link(\sigma, \mathcal{K})$, where the link corresponds to the subcomplex of $\mathcal{K}$ with the simplexes in $St(\sigma, \mathcal{K})$ without intersections with σ. The *join operator* $\mathcal{A} * \mathcal{B}$ works on two disjoint complexes $\mathcal{A}$ and

$\mathcal{B}$, and provides a new complex with the union of the vertices of each and combines their simplexes.

We can see the mentioned constructions above in image 2.6.

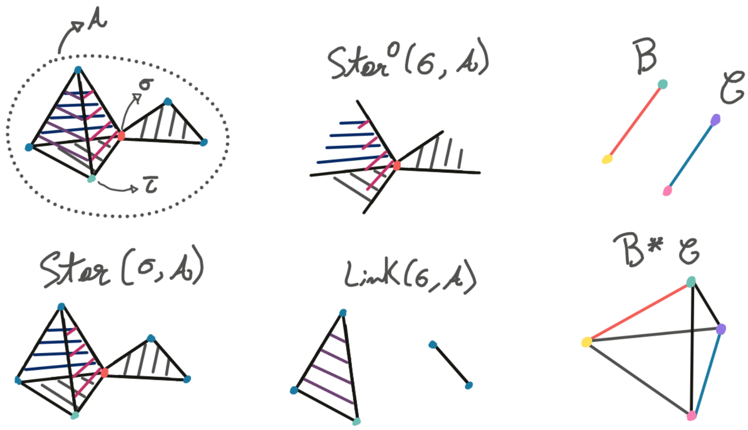


Figure 2.6 – Star, open star, link and join constructions from the simplicial complexes B and C.

2.1.3 Colorless tasks

A similar approach for tasks, part of the original work in distributed computing in Herlihy, Kozlov and Rajsbaum (2014), considers a relation between sets of input values and sets of output values in what are called colorless tasks, meaning that the identity of the agents does not matter in the context.

Each of the processes has access to its own identification, but (at least initially) not that of the other processes. They execute algorithms that can be represented as state machines, performing read and write operations in a shared memory in order to communicate with others, between their computation steps. A model called *immediate snapshot* describes the read and write steps as atomic and sequential, where they are always executed in adjacent steps and concurrently with other processes. This model allows for lower bounds, as an impossibility result in a stronger model can be easily extended.

In a given problem with unreliable communication, $n+1$ processes are considered to run a protocol (a program alongside the machine specifications), while at most n may crash silently. In the asynchronous case, fault detection is impossible, as one cannot distinguish between a slow process and a dead one.

An *execution* is a set of ordered configurations. Each *configuration* represents a collection of simultaneous process states, that is, the values held by each agent in a given moment. Distinguished configurations represent the *initial* and *final* sets of states for a given task. For example, in a binary consensus task, all combinations of states within the two possible values are acceptable in the initial configuration, meanwhile only the configurations where all processes possess the same state - 0 or 1 - are acceptable as final ones. A *step* is triple C_n, S_n, C_{n+1}, where C_n and C_{n+1} are the configurations at the beginning and at the end of the step, and S_n is the set of processes that participated in said step, and only they can change between C_n and C_{n+1}.

The formalization of a colorless task is composed of a *colorless input assignment* $\mathcal{I}$, some *colorless output assignment* $\mathcal{O}$ and a *relation* Δ that specifies which outputs are possible for each input assignment. Note that not all values within a colorless input assignment must be chosen by the processes of every configuration. Similarly, processes in a final configuration needs only to choose from the domain of output values. This can be illustrated again with the case of the binary consensus, where the following definition applies for any number of processes. First we have that all possible combinations are valid initially, but only configurations entirely composed of 0's or 1's are valid at the end.

$$\begin{aligned} \mathcal{I} = \quad & \{\{0\}\,\{1\}\,\{0,1\}\} \\ \mathcal{O} = \quad & \{\{0\}\,\{1\}\} \end{aligned}$$

And this relation is expressed with the following carrier map, where sets of states with values either 0 or 1 have to be mapped to the same configuration, and any other combination must be mapped to a single valued configuration.

$$\Delta(\mathbf{I}) = \begin{cases} \{\{0\}\} & \text{if I = } \{0\} \\ \{\{1\}\} & \text{if I = } \{1\} \\ \{\{0\}, \{1\}\} & \text{if I = } \{0,1\} \end{cases}$$

The protocol describing the behavior of the processes can be decomposed into two parts with distinct aspects of the problem taken into consideration in each. The first one is a *task-independent protocol*, related to the communication model, as each process updates its state based on the exchange, successful or not, with other processes. The second part is a *task-dependent decision*, which happens after enough rounds of communication have passed and a value is deterministically decided for each process. As it is a colorless task, the decision cannot depend on the process names, only in the set of states. A process is said to chose or decide the output value u from its final view v if $\delta(v) = u$, where δ is the decision map. The concept of reachable configurations comes from the different views that can be obtained from "perturbations" in the synchronization. Note that "perturbations" in the execution will change only the view of one process at a time.

A protocol is a triple $(\mathcal{I}, \mathcal{P}, \Xi)$, with the initial configurations $\mathcal{I}$, the reachable final configurations $\tau \in \mathcal{P}$ from the initial ones, and a map Ξ that carries each initial configuration to its set of reachable final configurations. A task is considered solvable by a protocol with a decision map δ, if for every configuration $\sigma \in \mathcal{I}$, and all reachable final configurations $\tau \in \mathcal{P}$, i.e. $\tau \in \Xi(\sigma)$, their decisions $\delta(\tau)$ are output assignments O that respect $\mathcal{O}$ given by the task's specification. That is, $O \in \Delta(\tau)$.

2.1.4 General tasks

Contrasting from the colorless tasks presented before, the general definition of tasks considers that the processes are identifiable and a solution may depend of their IDs. This allows for specific configurations to be described, in which combinations of processes must assume specified values. There are two main ways of defining general tasks for distributed computing, the first one was developed relying on carrier maps to represent the relationship between input/output/protocol complexes and the conditions for task solvability, established in Herlihy, Kozlov and Rajsbaum (2014). This can be found bellow.

Definition 2.5 *A general task is a triple* $(\mathcal{I}, \mathcal{O}, \Delta)$ *where:*

- $\mathcal{I}$ *is a chromatic input complex, it represents the possible initial configurations for the task;*
- $\mathcal{O}$ *is a chromatic output complex, it represents the valid final configurations for the task;*
- Δ *is a chromatic carrier map from* $\mathcal{I}$ *to* $\mathcal{O}$*, it shows what are the valid mappings that satisfy the task's objective.*

The vertices are colored with Π *and labeled with* V^{in} *and* V^{out}*, so that they are uniquely identified by their color and label.*

The possible solutions are depicted as *protocols*, as follows.

Definition 2.6 *A protocol for* $n+1$ *processes is a triple* $(\mathcal{I}, \mathcal{P}, \Xi)$*, where:*

- $\mathcal{I}$ *is a pure n-dimensional chromatic simplicial complex, it is the same presented in definition 2.5;*

- *$\mathcal{P}$ is a pure n-dimensional chromatic simplicial complex, it is the configurations of the systems after rounds of communications and its evolution represents the increased knowledge of the agents;*
- *Ξ is a chromatic strict carrier map from $\mathcal{I}$ to $\mathcal{P}$, where $\mathcal{P} = \bigcup_{\sigma \in \mathcal{I}} \Xi(\sigma)$, it represents the communication rounds of the protocol and how the simplices evolve through them.*

The vertices are as well colored with Π and labeled with V^{in} and V^{out}, also being uniquely identified by their color and label.

Definition 2.7 *The solvability of a general task $(\mathcal{I}, \mathcal{O}, \Delta)$ by a protocol $(\mathcal{I}, \mathcal{P}, \Xi)$ is defined by the existence of a chromatic simplicial map $\delta : \mathcal{P} \to \mathcal{O}$ that satisfies*

$$\delta(\Xi(\sigma)) \subseteq \Delta(\sigma) \textit{ for all } \sigma \in \mathcal{I}$$

This first approach is easier to visualize as all objects are the presented simplicial complexes and the arrows follow the sense of their evolution and growth. It can be seen in the diagram in diagram 2.1. The carrier map establishes relationships that are not functional, hence harder to rigorously define.

$$\begin{array}{ccc} \mathcal{I} & \longrightarrow & \mathcal{O} \\ \downarrow & \nearrow & \\ \mathcal{P} & & \end{array} \tag{2.1}$$

In the second approach all relations are functional, as chromatic simplicial complex maps. This is done by changing the direction of the arrows, where they now represent the existence of a unique “folding” of the complex into the one from the previous step. The task and protocol definitions change slightly, with the biggest differences seen in the task solvability criteria. An instance of this is seen in the definition of tasks via the dynamic epistemic logic point of view, in section 3.2.3.

Definition 2.8 *A general task is a triple $(\mathcal{I}, \mathcal{O}, t)$, where*

- *$\mathcal{I}$ and $\mathcal{O}$ are pure n-dimensional chromatic simplicial complexes, the same presented in definition 2.5*
- *$t : \mathcal{T} \to \mathcal{I} \times \mathcal{O}$ is a restriction function (chromatic simplicial map) onto the product of the input and output complexes. The resulting $(n \times n)-$ dimensional $\mathcal{T}$ chromatic simplicial complex represents the valid pairs of final configurations for each initial configuration.*

Note that we will always be able to project $\mathcal{T}$ into $\mathcal{I} \times \mathcal{O}$ via $\Pi_{\mathcal{I}}$.

The other formulation, with an approach based in category theory and first considered in Goubault (2021), is described as follows.

Definition 2.9 *A specification for $n+1$ processes is a tuple $(P, \mathcal{S})$ where*

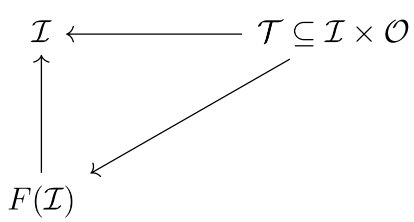
- *P is an endofunctor that operates on simplicial complexes, this is the protocol complex that represents the communications executed between the agents from the configurations in $\mathcal{I}$. It represents how to produce new states from a set of states after rounds.*
- *$\mathcal{S} : P \to id$ is a natural transformation between endofunctors, and it encodes the specification of the machine. It specifies the algebra on which the operations happen. It is a map of how to perform the color preserving folds of the complexes, dictating how to relate the states.*

Note that there will always be a map from $P(x) \to x$, i.e. the "evolved" input complex after a number of communication rounds can always be folded back into the original input complex through $\mathcal{S}$.

The original definition requires the protocol carrier map to be strict, i.e. $\Xi(\sigma \cap \tau) = \Xi(\sigma) \cap \Xi(\tau)$. The chromatic simplicial map that models it in $P(\mathcal{I})$ preserves this property as it maps the entire chromatic simplicial complex into another one, preserving labeling due to its chromatic nature and carrying simplicies to simplices by definition.

Given these elements, we now define the solvability of a general task, given a task specification and a protocol.

Definition 2.10 *The solvability of a general task $(\mathcal{I}, \mathcal{O}, t)$ by a protocol $(P, \mathcal{S})$ depends on the existence of a chromatic simplicial map δ from $\mathcal{A} \subseteq \mathcal{I} \times \mathcal{O}$, the task specification, to $P(\mathcal{I})$, the protocol complex applied to the initial configurations, such that the following diagram commutes.*



This definition raises the algebraic structured presented in section 2.4.5, where the protocol complex endofunctor may be operated via the natural transformation $\mathcal{S}$.

2.2 Look Compute Move

General notes follow with the taxonomy of the gathering and exploration problems over graphs using the Look-Compute-Move model.

2.2.1 LCM model

The Look-Compute-Move model is an interesting approach to discussing robotic operations, as it reduces any implementation to three steps that repeat in a loop. The first one is LOOK, where a robot observes its environment and corresponds to the activation of the available exteroceptive sensors. This is followed by a COMPUTE step, where the information gathered by the sensors is used by the algorithm and a decision is taken, which is then executed in the MOVE step, which corresponds to the activation of the actuators of the robot, where it changes its state.

Research done in this area targets the upper and lower bounds for the execution of missions, where the interest lies in finding what is the minimum necessary for a task to be possible or if the scenario available is sufficient. In the most general version of the model, the robots are considered identical, with the exact same deterministic program in all of them. Beyond that, there's no identification of the robots, being impossible to distinguish them only with their appearance. They also have no sense of orientation, with no shared labels for vertices or edges. They occupy no space and can coexist in the same vertex, but it is impossible for them to see how many others are present with them.

In the case of luminous robots, the robots' states are composed of their state machine with and added integer value associated with its light. When the LOOK action is performed, it is only capable of detecting the existence or not of a robot at a certain vertex with a specific color.

Usually and in this case, robot are oblivious, meaning that they only have memory of their most recent past state. A configuration of the system is a vector with the state of the robots. The possible actions are only LOOK or MOVE, as COMPUTE is assumed to always happen atomically after a LOOK. It is also assumed that the algorithm always terminate, where all robots always reach a decision given a state. The overall functioning is described in algorithm 1.

Algorithm 1 A general LOOK-COMPUTE-MOVE algorithm. Code for robot p_i.

function $\mathcal{A}(G, v_i)$
    while true **do**
        $view_i \leftarrow$ LOOK(G)
        $(v_i, r_i) \leftarrow$ COMPUTE$(view_i)$
        MOVE(v_i, r_i)
    end while
end function

Where v_i is the targeted vertex and r_i the chosen color for robot i after perceiving $view_i$ from the graph.

2.2.2 Taxonomy

In order to represent the most varied scenarios in the real applications, the LCM model has many variations that aim to modify the capabilities of the robots and characteristics of the environment. They change the amount of information perceived by the agents, what is observable of the world and how can the robots interact.

Regarding the robots, they may be understood either as physical agents in an Euclidean space or as agents in a graph. In both cases, there are some properties that we need to observe in order to classify the scenario.

- Movement: robots can have a source of randomness in their input, making the way they move *random* or *deterministic*.
- Identity/Symmetry: they may be *labeled* or *anonymous* / *symmetric* or *asymmetric*, corresponding to the their distinguishability.
- Knowledge: what do they know of the *map* and *other agents* considerably influences their algorithms, which is related to the information accessible in preparation of a mission.
- Communication: there are multiple different means of exchanging information and their capacity of sending *self-generated messages*, which is the source of most of the difficulties in asserting the acquired knowledge throughout the mission.
- Chirality: it is possible for the robots to agree on a *sense of rotation*, also influencing their capacity to make coordinated decisions.

- Collisions: they may have different interpretations and consequences if they are either *immaterial*, *fail-stop* or *intolerable*.
- Movement: regarding the self knowledge of the robots, *rigid*, *fixed* or *non-rigid* movements determine the garantees of MOVE operations, such as if they always go to the desired destination, cover a minimum distance or none of them.
- Space and visibility: someones we have to consider the *extent* agents may have, and if they limit each other's visibility with *opacity*.

When considering the environments in which the robots operate, we will be considering only the discrete case, where we use approximations of the real world in graphs. Again, there are specifications that dictate what the agents can perceive and what should they expect during navigation, here we see the most important ones.

- Identity/Symmetry: the nodes may have *distinct identities*, which heavily influences the ease of localization and coordination.
- Edge labeling: similarly to the case above, edges may be *locally* or *globally* consistent, being also used as part of the efforts to break symmetry in the mission.
- Time: different schedulers exists with varying levels of guarantees, while centralized ones may keep everyone *synchronous*, weaker constraints exist with *semi-synchronous* and *asynchronous* operations.
- Storage: this is an aspect of communication that depends on the environment, where it allows for *tokens* or *messages* to be left by one agent for another.
- Agent generation: it may be possible for agents to be cloned into new ones or existing ones merge, such as with the activation or synchronization of robots during a mission.
- Fault tolerance: the agents and the scheduler may be subject to *crash faults* or *byzantine faults*, again changing the assumptions of the mission, being relevant to know if it is possible to detect them or keep working after they appear.

2.3 Modal logic

The notes on the basis of modal logic for this work are mostly based from Benthem (2010).

2.3.1 Introduction and syntax

Modal logic constitutes a super-set of the standard formal logic, with added modal operators that allow for the expression of notions similar to "possibility" and "necessity", which use as notation the symbols $\Box$, read as "box" and $\Diamond$, read as "diamond". Those additions allow us to create formulas composed of propositions, statements that can be either true or false, with added meaning, such as "*p* will *eventually* become true" and "*p always* is true throughout some relevant range of situations".

Multiple interpretations exist of modal logic, interesting examples are Godel's view of $\Box p$ as the "mathematical provability of p" and Tarski's view of modal formulas as describing subsets in topological spaces. Another of its crossings to different fields is Montague semantics, where intentional expressions are studied with a mix of modal logic and type theory. The idea itself of modal logic being an enrichment of classical logical is not agreed by all, and a common perspective is of it as "local" quantifiers that refer to objects "accessible" from the current one, which will be explored a bit with the possible worlds model in section 2.3.2.

The first step for a formalization is the definition of the atomic elements

$$AT = p, q, r, ..., \top, \bot$$

with generic atoms, an atom representing "always true" and another one representing "always false", respectively.

This is then followed by a way to construct formulas from those atoms

$$\varphi ::= AT \mid \neg\varphi \mid (\varphi \land \psi) \mid (\varphi \lor \psi) \mid (\varphi \to \psi) \mid \Diamond\varphi \mid \Box\varphi$$

From this, the first concept to be understood is that all atoms are formulas, and if an atom is a formula, then so is its constructions as $\neg\varphi$, $(\varphi \land \psi)$, which continues recursively. With this, all formulas can be defined in a finite number of steps. Examples of equivalent expressions are:

1. $\Diamond\varphi \leftrightarrow \neg\Box\neg\varphi$
2. $\Box\varphi \leftrightarrow \neg\Diamond\neg\varphi$
3. $\neg\Box(\Diamond\varphi \land \Diamond\neg\varphi) \leftrightarrow \Box\Diamond\varphi \to \Diamond\Box\varphi$

The first one can be read as "it is possible that φ is true if and only if it is not necessary that φ is not true", and the second as "φ is necessarily true *iff* it is not possible that φ is not true". The third one is a formalization of the idea "nothing is absolutely relative", with the possible interpretation of the first half as "it is not necessary that the possibility of φ coexists with the possibility of $\neg\varphi$", and the second half is its rewriting into McKinsey axiom, an important conclusion in modal logic that will not be explored here.

In our case, this formalism will be another point of view on the same constructions used to represent distributed executions of tasks. Some usages of this equivalence will be shown later in this work. It is useful to keep in mind that the different interpretations that can be given to the same structures present here, in distributed computing and robotics.

2.3.2 Possible worlds model

An important usage of the modal logic system is the multiple worlds model, in which we give a semantic interpretation to the formulas as a graph-like structure that represent the possible truth values for the propositions involved in a given formula. It consists of a specific interpretation using Kripke models.

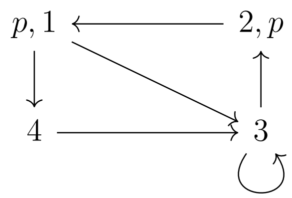
The possible worlds model is defined as a triple $M = (W, R, V)$, composed of a collection of worlds W, an accessibility relation between worlds R, and a valuation map with truth values V for a proposition p at the world s. The relation R expresses the constraints in transitioning between worlds, while the complete graph may display the possible evolution from an initial proposition. The truth values obtainable from $V(p, s)$ can also be expressed as $\mathbf{M}, s \models \varphi$, whenever φ is true in a given state s of the model $\mathbf{M}$.

The truth definition in the possible worlds models happens as follows

1. $\mathbf{M}, s \models p$ iff $V(p, s) = 1$
2. $\mathbf{M}, s \models \neg\varphi$ iff not $\mathbf{M}, s \models \varphi$
3. $\mathbf{M}, s \models \varphi \wedge \psi$ iff $\mathbf{M}, s \models \varphi$ and $\mathbf{M}, s \models \psi$
4. $\mathbf{M}, s \models \Box\varphi$ iff for all inciding t of s as: $\mathbf{M}, t \models \varphi$
5. $\mathbf{M}, s \models \diamond\varphi$ iff for some inciding t of s as: $\mathbf{M}, t \models \varphi$

Here, 1 is the basic definition of the valuation map, 2 and 3 expand this definition to the negation and conjunction. 4 says that for the valuation of φ in the world s in the model $\mathbf{M}$ to be necessarily true, it has to hold in *every* world t of the same model that leads to s. 5 says that for possibility of φ being true when valuated in a similar situation, *some* world t of the same model that leads to s has to be true. Finally, the concept of modal validity can be affirmed to a given formula φ if $\mathbf{M}, s \models \varphi \; \forall(\mathbf{M}, s)$, also written as $\models \varphi$.

Having those definitions, we can profit of the possible worlds diagrams to both have a visual representation of our formulas and to infer the possible values of each "node" from the relationship graph of an existing system. A way of using the latter is finding where a given modal formula holds in the analyzed graph, such as



(2.3)

In the case that wish to know in which worlds the formula $\Diamond\Box\Diamond p$ holds, we can use the truth value definitions in the graph 2.3.2 to expand the known information, giving us the following (non exhaustive) valid formulas after some iterations:

- 1: $\Diamond\Box\Diamond p$
- 2: $p, \; \Diamond p$
- 3: $\Diamond p, \; \Box\Diamond p, \; \Diamond\Box\Diamond p$
- 4: $\Box\Diamond p, \; \Diamond\Box\Diamond p$

Note that the values *p* are true in 1 and 2 since the beginning of the evaluation. As it is impossible to conclude $\Diamond\Box\Diamond p$ for the world 2, no matter how many more iterations, we cannot say that it is a valid formula, i.e. it is not true in all worlds of the model. However, we have arrived to the fact that the modal formula holds for the worlds 1, 3 and 4.

Those examples only make use of the base formal system of modal logic, however it is important to note that they are compatible with the temporal-epistemic extensions and that integration is relevant for the envisioned work, even though they are not discussed here.

2.3.3 Temporal logic

The first extension to logic that suits our needs increasing our capacity of formally representing robotics tasks is temporal logic. It is capable of placing propositions in the past or future, with certainty or not, either as a finite event or a continuous one. The formalization starts with the following addition to the syntax rules and interpretations

$$\varphi ::= ... \mid F\varphi \mid P\varphi \mid G\varphi \mid H\varphi$$

where $\mathbf{F}$ and $\mathbf{P}$ are initially defined as the idea of "at least once in the future, φ will be true" and "at least once in the past, φ has been true", respectively. Those two operators are derived into the universal modalities $\mathbf{G}$ and $\mathbf{H}$, that inform that "always in the future, from now, φ is true" and "always in the past, until now, φ was true".

From this initial specification some interesting expressions can already be formed, such as:

1. $\mathbf{GF}\varphi$
2. $\mathbf{PH}\varphi$
3. $\mathbf{G}(\varphi \rightarrow \mathbf{F}\varphi)$

Expressions 1 and 2 join the base operators with universal modalities, generating finite events that are sure to happen at some point, being them read as "φ is always going to be true at some later stage" and "once upon a time, φ had always been true", respectively. Example 3 can be read as "φ will always imply that φ will be true in the future", that can also be understood as φ being true will always "enable" itself afterwards.

Those constructions also enable the interpretation of validity throughout time across worlds (possibilities of a system, shown in section 2.3.2), such as:

1. $\mathbf{M}, t \models \mathbf{F}\varphi$ *iff* for some $t' > t : \mathbf{M}, t' \models \varphi$
2. $\mathbf{M}, t \models \mathbf{P}\varphi$ *iff* for some $t' < t : \mathbf{M}, t' \models \varphi$

which shows how the ordering of moments in time "before" or "after" others may be used for creating expressions in this extension.

Similarly to what will be presented on the following section on epistemic logic, there are further tools available that allow for the deduction and extraction of truths about the scenario that are fundamental in guaranteeing the behavior of some studied system.

2.3.4 Epistemic logic

The epistemic extension to modal logic deals with knowledge, what information is shared and what can be assumed from what each one has access to. With it, the ideas of an agent having a certain information or something being common knowledge among a group of agents become formalized through the following increased syntax rules

$$\varphi ::= ... \mid K_i\varphi \mid C_G\varphi$$

where $K_i\varphi$ expresses that the agent i knows φ to be true, and $C_G\varphi$ that the group of agents G knows φ to be true, and each agent knows that the other ones know as well. It is interesting to note that the common knowledge operator can represented in the nested form C^k, such as in the following case with the two agents g and w:

$C^2_{\{g,w\}}\varphi \rightarrow K_g(K_g\varphi \wedge K_w\varphi) \wedge K_w(K_g\varphi \wedge K_w\varphi)$

The idea of an agent i considering the possibility of some information φ has the special notation of $B_i\varphi$, being it is equivalent to $\neg K_{i\ \neg\varphi}$. The latter can also be read as "the agent i does not know if φ is false".

Here are some examples of how this augmented syntax can be used:

1. $\neg K_{Q\ \varphi} \wedge \neg K_{Q\ \neg\varphi}$
2. $B_Q(K_{A\ \varphi} \vee K_{A\ \neg\varphi})$

The first example models the idea that Q does not know weather φ is true, a translation from the more immediate reading of "Q doesn't know if φ and it also doesn't know if $\neg\varphi$". The second example makes use of the "belief" notation, being possible to read it as "Q thinks that A might know φ".

This extension is relevant for the study of distributed robotic tasks, where the possible knowledge of the individual robots is fundamental for evaluating the feasibility of those tasks.

2.3.5 Kripke frames and models on epistemic logic

The idea of possible worlds comes from Kripke models, where a layer of semantics is added on top of a Kripe frame, in order to represent the accessibility relations as equivalences of worlds between agents. The are multiple systems of interpretations for those models, and one of them that interests us is $\mathbf{S5_n}$, where the worlds and their connectiosn represent knowledge of the environment by the agents.

Definition 2.11 *A* Kripke frame *on the epistemic system* $\mathbf{S5_n}$ *is a pair* $M = \langle S, \sim\rangle$ *defined over a set* A *of agents, with a set of states* S *and a family of binary equivalence* *relations between states for every $a \in A$. It is written $u \sim_a v$ for the worlds u and v that are indistinguishable for agent a.*

A Kripke frame is considered *proper* if every pair of states can be distinguished by at least one agent, that is, no two worlds can be merged without loss of information.

2.3.6 Dynamic epistemic logic (DEL)

The study of change is possible within the framework provided by epistemic logic through the extension with action models. This is particularly useful when dealing with message passing systems, where decisions are made by an agent, which in turn affect the other agents. We add the notion of an action c^b, composed of a series of communications c between the agents and their starting values $\{b_0, b_1, ..., b_i\}$. In the case of two agents g and w, we represent with the 3 digraphs in figure 2.7 the situations where both communicate or at most one fails. Note that it only encodes the communications, without its content.

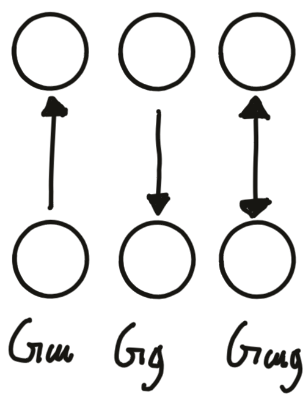


Figure 2.7 – Digraphs representing the scenarios where both process successfully exchange information and when at most one of them fail in the communication. One of these scenarios is executed at each communication round.

An epistemic model M, which already expresses the shared knowledge of the agents is accompanied by an action model $\mathcal{A}$, with the definition bellow.

Definition 2.12 *An action model $\mathcal{A}$ is a triple $\langle T, \sim, pre \rangle$, where:*

- *T is the domain of action points*
- *$\sim_a$ is an equivalence relation on T, for each agent $a \in A$*
- *$pre : T \to \mathcal{L}_K$ is a function that assigns a precondition formula $pre(t)$ for all $t \in T$.*

Given an action model, the indistinguishability relation generates a Kripke model similar to the epistemic model, where $t \sim_a t'$ if and only if they have the same view, i.e. $view_a(t) = view_a(t')$. The view their current knowledge of the world, gained from the rounds of communication.

The epistemic and action models are combined into the *product update model* with the possible worlds consequent of each action in $M[\mathcal{A}]$.

Definition 2.13 *A product update model $M[\mathcal{A}] = \langle S[\mathcal{A}], \sim^{[\mathcal{A}]}, L[\mathcal{A}]\rangle$ is the combination of an epistemic model M and an action model $\mathcal{A}$, where:*

- *The worlds in $S[\mathcal{A}]$ are pairs (s, t), such that $pre(t)$ holds in s*
- *Two worlds $(s, t) \sim_a^{[\mathcal{A}]} (s', t')$ if $s \sim_a s'$ and $t \sim_a t'$*
- *The valuation of $L[\mathcal{A}]((s, t)) = L(s)$*

2.4 Category theory

Here, the notes on category theory for the study of robot tasks and their connection with distributed computing and modal logics are shown, primarily studied with Riehl (2016), Smith (2018).

2.4.1 Introduction

Category theory came to existence as a set of formalizations allowing the study of generic abstract structures in mathematics, following the efforts of algebraic topology, being first proposed by S. Mac Lane in Lane (1971). The key take from the categorical view is that *relations* are more important than *elements* to describe structure. The three main concepts are categories, functors, which allow for the transformation from one category to another, and natural transformations, that allow for the transformation from one functor to another.

Much of the interest in category theory comes from the fact that it reflects on itself, meaning that all constructions when viewed from the proper perspective give rise to their own category, such as functors between categories creating the functor category. This multitude of perspectives can be evaluated with varying levels of equivalence (with a corresponding rigor), such as equality, isomorphism, equivalence, natural isomorphism and adjunction. In the same line of thought, it is also useful how that reflexivity allows for the perception of universal constructions, phenomena that are present throughout the different categories, each with their own interpretation but sharing the same properties, such as limits, colimits, adjunctions and ends. Those constructions will not be covered here but it must be noted the richness in theoretical representations.

2.4.2 Categories

The first relevant definition is that of categories, which may be understood as families of structures with structure-preserving maps between them.

Definition 2.14 *A category $\mathcal{C}$ is composed of two kinds of things:*

- *$\mathcal{C}$-objects, usually expressed with upper case letters;*
- *$\mathcal{C}$-arrows, also called $\mathcal{C}$-morphisms, usually expressed with lower case letters.*

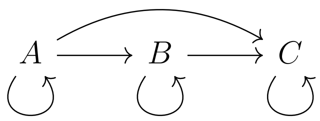
Where the objects must respect three axioms:

1. Sources and targets. *For each arrow f, a specific collection of objects is associated with its domain and codomain, those not being necessarily distinct. We may write $f : A \to B$ for the arrow f with $src(f) = A$ and $tgt(f) = B$, or $dom(f) = A$ and $cod(f) = B$.*
2. Composition. *Any pair of arrows $f : A \to B$ and $g : B \to C$, where $dom(g) = cod(f)$, may be composed into a new arrow $g \circ f : A \to C$. This is understood as the application of f followed by the application of g in the previous result, and is called the* composite *of f and g.*
3. Identity arrows. *Any given object A is equipped with an identity arrow $1_A : A \to A$, that has $dom(1_A) = cod(1_A)$.*

And the arrows must respect two other axioms:

1. Associativity of composition. *The composition operation is associative, which means that for any $f : A \to B$, $g : B \to C$, $h : C \to D$, we will always have that $h \circ (g \circ f) = (h \circ g) \circ f$.*
2. Identity arrows behave as identities. *The identity arrows can always be added to operations without change in result, such that for any arrow $f : A \to B$, we have that $f \circ 1_A = f = 1_B \circ f$.*

Those properties are exemplified in the following commutative diagram, where paths that start and finish on the same objects are always equivalent.



(2.4)

As mentioned before, categories may be used to represent an infinity of mathematical structures, table 2.1 shows some categories with their respective interpretations of the main elements. It is interesting to note that in this list there are algebraic categories (Set, Grp, Rng, Bool), order categories (Pos, Tot), geometric categories (Top, Met, Vect$_k$) and a logic category (Proof$_T$).

Table 2.1 – Some examples of notorious categories of well known mathematical structures.

Category	Structure	Objects	Morphisms
Set	Sets	sets	total functions
Grp	Groups	groups	group homomorphisms
Ab	Abelian groups	abelian groups	group homomorphisms
Rng	Rings	rings	ring homomorphisms
Bool	Boolean algebras	Boolean algebras	structure preserving maps
Pos	Partially-ordered sets	sets	order preserving maps
Tot	Totally-ordered sets	sets	order preserving maps
Top	Topopologies	topological spaces	continuous maps between spaces
Met	Metric spaces	set of points S equipped with a real metric d	non-expansive maps
Vect$_k$	Vector spaces over a field k	set of vectors equipped with addition and multiplication by scalars in the field k	linear maps between spaces
Proof$_t$	Formal theory T	sentences φ, ψ, ... of the formal language T	the target of the arrow is sufficient to prove the source

2.4.3 Duality

A useful property of categorical constructions is the immediately available dual interpretation. It corresponds to the $\mathcal{C}^{op}$ category associated with $\mathcal{C}$, where the following happens:

1. $\mathcal{C}^{op}$ -objects are the same as $\mathcal{C}$ -objects.
2. Given the arrows in $\mathcal{C}$, such as $f : A \rightarrow B$, the $\mathcal{C}^{op}$ will have the source and target swapped, such as $f^{op} : B \rightarrow A$.
3. The identity arrows are preserved the same, $1_A^{op} = 1_A$.
4. Composition also has the source and target swapped, such that $f \circ^{op} g = g \circ f$;

This way of constructing categories allows us to have "free proofs", from whatever categorical concept.

2.4.4 Functors

Functors appear as the first level of higher-order abstraction, as they represent transitions between two categories. It is attributed to John Baez, a researcher in higher-order category theory, the quote "every sufficiently good analogy is yearning to become a functor.". This expressed the fundamental functioning of functors as ways to transfer acquired notions from one perspective into another.

Functors are one of the structure-preserving maps mentioned at the beginning of the section, and given two categories $\mathcal{C}$ and $\mathcal{D}$, the functor $F : \mathcal{C} \rightarrow \mathcal{D}$ satisfies the following attributes:

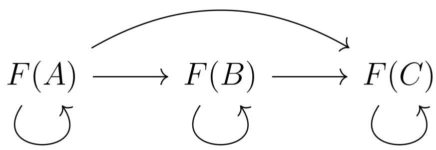
(1) $\mathcal{D}$ -objects are obtainable from $\mathcal{C}$ -objects through F, expressed as $F(A) = B$, belonging to the respective categories.

(2) $\mathcal{D}$ -arrows between $\mathcal{D}$ -objects are preserved through F, such that the arrow $F(f) : F(A) \rightarrow F(B)$ in $\mathcal{D}$ is equivalent to the arrow $f : A \rightarrow B$ in $\mathcal{C}$.

These operations via the functor F must respect the following conditions:

1. *Preserve identities*. Any identity arrow for a $\mathcal{C}$ object A will be mapped to the corresponding identity in $\mathcal{D}$, such that $F(1_A) = 1_{F(A)}$.
2. *Respect composition*. Any two arrows f and g in $\mathcal{C}$, with the corresponding composite $g \circ f$, with always have $F(g \circ f) = F(g) \circ F(f)$.

Those properties are expressed in the following diagram 2.4.4, which is the same as 2.4.2 after having a functor F applied. It always has similar paths commuting.



(2.5)

2.4.5 Algebras

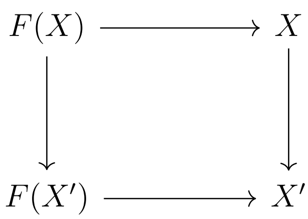
It is possible to describe an algebraic structure from endofunctors, that is, functors that map objects and morphisms inside a single category.

Definition 2.15 *Given an endofunctor $F : \mathcal{C} \rightarrow \mathcal{C}$, an F-algebra is a pair $(X, a : F(X) \rightarrow X)$ where X is an object of $\mathcal{C}$.*

This definition lets us define the interaction of algebras in a category, as follows.

Category 2.1 *The category of F-algebras $Alg(F)$ is composed of*

- *objects: algebras of a given endofunctor $F : \mathcal{C} \rightarrow \mathcal{C}$*
- *morphisms: given two F-algebras (X, a) and X', a', there is a morphism between them iff there is a morphism from X to X' in $\mathcal{C}$ and the following diagram commutes*



Mathematical objects that fit that structure may be manipulated in more interesting ways. This is part of the future ideas for the structures used in the formalization of robot tasks, leading to the co-existing modalities of temporal and epistemic logics.

3 CONNECTIONS BETWEEN LOGIC AND TOPOLOGY

The first studied connection is made between the topological structures called simplicial complexes, used in distributed computing, and Kripke frames, first created as a semantic system for modal logics. An important theorem in modal logics will be presented in section 3.1, alongside some considerations on its usage for the topological applications. It is then followed by a work that makes use of this equivalence will be presented in section 3.2. The equivalence between those two concepts will be shown in the categorical perspective can be found in the appendix 7.1,

3.1 Knowledge Gain Theorem

An useful property observed in modal logic systems, such as $S5_n$, and extended for simplicial models like the one studied above, is the guaranteed persistence of knowledge, and the certainty that any observed information in a state will always be present in the states leading to it. This is formalized in the *Knowledge Gain Theorem*, as follows:

Theorem 3.1 *Consider simplicial models $M = \langle C, \mathcal{X}, l \rangle$, and a morphism $f : M \to M'$. Let $X \in \mathcal{F}(C)$ be a facet of M, a an agent, and ϕ a formula which does not contain negations except, possibly, in front of atomic propositions. Then, $M', f(X) \models \phi$ implies $M, X \models \phi$.*

This can be proven by induction. Considering first atomic propositions, this holds as morphisms preserve valuation, that is, $M', f(Y) \models p$ iff $M, Y \models p$. Given the theorem true for atomic propositions, the case of conjunction and negation follow trivially by the induction hypothesis on ϕ.

For the knowledge operator, we suppose that $M', f(X) \models K_a\phi$, for us to show that $M, X \models K_a\phi$. If we consider another facet Y in which $a \in \mathcal{X}(\mathcal{X} \cap \mathcal{Y})$, we have that for any a-colored vertex v, $f(v) \in f(X) \cap f(Y)$, $\mathcal{X}(f(v)) = a$ and therefore $a \in \mathcal{X}(f(X) \cap f(Y))$ as defined in chromatic simplicial maps. The unchanged colouring within the intersection shows that $M', f(Y) \models \phi$, and with the induction hypothesis we have $M, Y \models \phi$.

For the common knowledge operator, we have that $M', f(X) \models C_B\phi$, for us to show that $M, X \models C_B\phi$. That can be shown in similar fashion as in the knowledge operator, where this has to be proven for all facets Y that are reachable from X from simplexes sharing a B-colored vertex. As any $f(Y)$ will be reachable from $f(X)$, so will $M', f(Y) \models \phi$ and therefore $M, Y \models \phi$.

This result is interesting as it gives us another interpretation for what happens in the communication rounds of a distributed system. We know that throughout the folding of the subdivision steps, from the protocol complex back to the input complex, the knowledge of the agents will be preserved, i.e. no knowledge is added to the model. The implications of this are still to be studied, and possibly stronger results may be achievable from different modal systems.

3.2 Approximate agreement from the epistemic logic perspective

An interesting usage of this shared concepts happens with the article Armenta-Segura, Rajsbaum and Ledent (2020), where the simplicial models seen before are used to represent the evolving knowledge of agents and a formalism is presented for thinking about the solvability of their tasks. The logic view on consensus has been first studied in Fagin (1995), and now we see it a relaxed version of the problem.

3.2.1 Approximate agreement

Approximate agreement is a widely worked problem in distributed computing, agents have to decide on values that are sufficiently close to each other, selecting them from an initial set of values. This is sometimes the case where consensus is impossible, and a relaxed result may be sufficient, such as when sensors need to estimate some measurements or clocks have to be synchronized. The closeness factor is parametrised by $\epsilon \geq 0$, where all agents have to reach values at most ϵ away from the others.

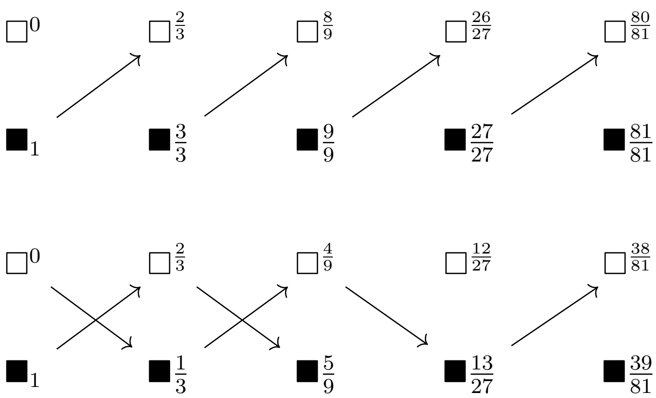
An example task is the requirement that two agents with arbitrary initial values decide in a pair (d_a, d_b) inside the interval of the initial ones with a distance of at most $1/3^k$ from one another, where k is the number of rounds executed. Algorithm 2 is a very simple solution to this, considering cases where messages may be lost, and $\perp$ is received, representing the absence of new information.

Here, the decision made by a process will depend on whether or not it receives some value from the other process. The reason for the failure does not matter and a process that fails to send in one round may send in another. We can see in figure 3.1

Algorithm 2 Approximate Agreement for $N = 2$ processes and $\epsilon = 1/3^k$. Code for process q_i.

```
function APPROXIMATEAGREEMENT(l)
    send(l)
    m ← receive()
    if m ≠ ⊥ then
        l ← l/3 + 2m/3
    end if
    return l
end function
```

two possible executions of algorithm 2 during 5 rounds, where all communications are depicted with arrows and the current value next to the agent.



(3.1)

3.2.2 Simplicial model for DEL

The simplicial model for DEL, first seen in section 2.3.6, is another usage of the equivalence seen in section 7.1.2. The simplicial model is maintained for the similar epistemic approach, and a *simplicial action model* is defined, which allows for the later definition of the knowledge of the agents gained from the execution of an algorithm.

Definition 3.1 *A simplicial action model is a triple* $\langle T, \mathcal{X}, pre\rangle$, *where*

- $(T, \mathcal{X})$ *define a chromatic simplicial complex, where the facets are communicative actions*
- *each facet* $X \in \mathcal{F}(T)$ *has pre-assigned a precondition formula* $pre(X) \in \mathcal{L}_K$

The following *product update simplicial model* is simply a simplicial model where the underlying complex is a subset of $C \times T$, the product of the simplicial epistemic model and the simplicial action model. The associated action model will only be available in the product if the preconditions allow for the connection to exist. Given an action $t = c^{b_0 b_1}$ between two processes $\{g, w\}$, a precondition $pre(c^{b_0 b_1}) = input_g^{b_0} \wedge input_w^{b_1}$, will guarantee that the values of the processes are indeed b_0 and b_1.

Definition 3.2 *A product update simplicial model $C[\mathcal{A}]$, given a collection of agents A, simplicial epistemic model C and a simplicial action model $\mathcal{A}$, and will have*

- *vertices in the form of $\{\langle a, view_a(c^{b_0 b_1})\rangle | a \in A, c^{b_0 b_1} \in calA\}$;*
- *facets in the form of $\{\langle a_0, view_{a_0}(c^{b_0 b_1})\rangle, \langle a_1, view_{a_1}(c^{b_0 b_1})\rangle\}$ for each action $c^{b_0 b_1} \in \mathcal{A}$;*
- *precondition that the agents involved have the input values of the actions as their current values.*

3.2.3 Tasks

The distributed computing perspective is seen again with the evolving knowledge of the agents via the product update simplicial model. This allows for the reasoning of tasks slightly differently than how it will be done in section 4.1.4. Here, we also define an input simplicial model $\mathcal{I} = \langle T, \mathcal{X}, l\rangle$, where each facet, alongside its labeling, is a possible initial configuration. Now we have two different simplicial action models: a communication model $\mathcal{T} = \langle T, \mathcal{X}, \mathsf{pre}\rangle$, representing the expected behavior of knowledge gain between the agents over time, and an algorithm $\mathcal{A}$, that encodes the actual behavior of the agents in the possible executions from the initial configurations.

A task $\mathcal{T}$ is solvable via the algorithm $\mathcal{A}$ if each agent produces an output corresponding to a facet of $\mathcal{T}$ via $\mathcal{I}[\mathcal{A}]$, respecting the preconditions. A morphism δ must exist, mapping facets $X \in \mathcal{I}[\mathcal{A}]$ to facets $(i, dec) \in \mathcal{I}[\mathcal{T}]$, where dec is a facet of $\mathcal{T}$ with the decisions of the agents in X (and i is the shared facet from $\mathcal{I}$), so that $pre(dec)$ also holds in i. Equivalently, we say that the following diagram in diagram 3.2 must commute. The simplicial morphism $\Pi_{\mathcal{I}}$ is a simple projection from the subset of the products $\mathcal{I} \times \mathcal{T}$ and $\mathcal{I} \times \mathcal{A}$.

$$\begin{array}{ccc} & & \mathcal{I}[\mathcal{A}] \\ & \swarrow & \downarrow \\ \mathcal{I} & \longleftarrow & \mathcal{I}[\mathcal{T}] \end{array} \tag{3.2}$$

3.2.4 Approximate agreement as a task

Consider the N-approximate agreement task, which requires that the decided values d_a, d_b after k rounds respect $|d_a - d_b| \leq 1/N$, and they have to be of the form k/N, for $0 \leq k \leq N$. This can still be used to solve the continuous approximate agreement as we just need to chose N so that $1/N < \epsilon$. Now we will express the problem of N-approximate agreement as a task.

The input simplicial model $\mathcal{I}$, the task $\mathcal{T}$ and the simplicial product update model for $N = 5$ rounds are seen in image 3.1. They correspond to the possible initial configurations with two agents that have to chose between the values $\{0, 1\}$ and end within $= 1/5$ of one another. The values in the product are their initial values in $\mathcal{I}$, each "block" is the possible decisions that respect $\mathcal{T}$. The two edges in the middle connecting the two "blocks" represent the indistinguishability of the initial values, as they are not only connected to the corresponding subdivisions, but to the other initial values, similarly to what is displayed in $\mathcal{I}$.

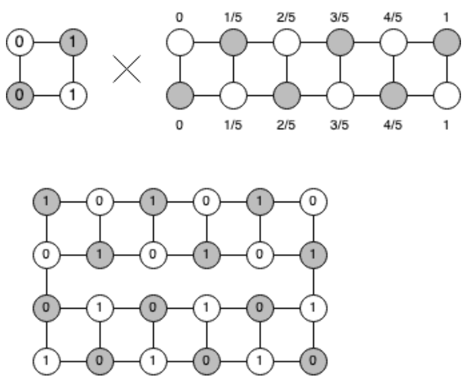


Figure 3.1 – $\mathcal{I}$, $\mathcal{T}$ and the subset of their product relevant for the approximate agreement task definition.

Now, if we consider again the dynamical network model $\mathcal{DN}^r$, where processes may send or not messages to each other at each round, we have the update model presented in image 3.2.

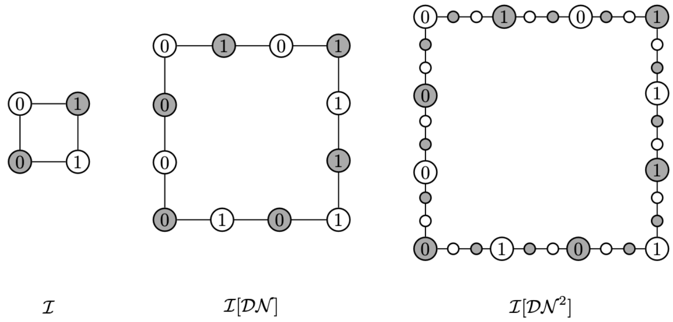


Figure 3.2 – Evolution of the knowledge in the $\mathcal{DN}^r$ model, where each configuration is split in three with the partial or complete transmission of the messages. Credit to Armenta-Segura, Rajsbaum and Ledent (2020).

This leads us to theorem 3.2 bellow.

Theorem 3.2 *The N-approximate agreement task is not solvable in the r-round model $\mathcal{DN}^r$, when $N \geq 3r + 1$.*

Each configuration edge generates three other ones after a round, leading to 3^r edges after r rounds.

This means that, if $k = \lceil 3^r/2 \rceil$, an edge where $\varphi 01$ is true will be at most distance $k - 1$ from an edge of inputs 00 or 11. This says that the group knowledge of $\varphi 01$ cannot be nested more than $\lceil 3^r/2 \rceil$ times. If we choose to have $\varphi = E^k \varphi 01$ with $k = \lceil N/2 \rceil$, the same will be true for $N \geq 3r + 1$ as it implies that $\lceil N/2 \rceil \geq \lceil 3^r/2 \rceil$, hence it will not hold in any world Z of $\mathcal{DN}^r$.

Now, if we take the fact that 11 and 00 from $\mathcal{I}[\mathcal{T}]$ are both in the image of δ, because of diagram 3.2, and also that the image of $\delta(\mathcal{I}[\mathcal{DN}^r])$ will be connected, since $\mathcal{I}[\mathcal{DN}^r]$ is connected. At least one of the worlds between 11 and 00 should also be present in the image of δ, any of which would satisfy $E^k \varphi 01$, for $k = \lceil N/2 \rceil$.

This leads us to a situation where $\varphi := E^k \varphi 01$ does not hold Z in $\mathcal{I}[\mathcal{DN}^r]$, but does in $\delta(Z)$. However, we know that this is impossible due to theorem 3.1, showing that there is no map δ that solves N-approximate agreement in $\mathcal{DN}^r$, when $N \geq 3r + 1$.

4 DISTRIBUTED COMPUTING APPROACH FOR LCM

The publication of Alcántara et al. (2019) opened the usage of topology as a mathematical framework in dealing with robotics through the Look-Computer-Move model. This chapter is dedicated to the study of this intersection, starting with an in-depth review of the base work in section 4.1. Section 4.2 will discuss the usual scenarios of exploration missions with limited visibility. Lastly, section 4.3 will expose thoughts on the formalization of a simple robotic missions.

4.1 The topology of look-compute-move robot wait-free algorithms with hard termination

This work formalizes the connection of the asynchronous luminous robot (ALR) model on graph networks and asynchronous wait-free multiprocess read/write shared memory (WFSM) model. This relation of a Look-Compute-Move (LCM) robot with finite cycles with distributed computing is then used to provide a topological characterization of the robot tasks, as in Alcántara et al. (2019).

4.1.1 ALR, EALR, WSFM and their connections

4.1.1.1 Asynchronous luminous robot model

This model considers the existence of $N \geq 2$ autonomous, deterministic mobile robots, that take positions in the vertices of a graph $\mathcal{G}$. As it is asynchronous, each robot has its own notion of time and move independently at different speeds, with a scheduler deciding the next operation to be executed and by who. The robots are equipped with a constant number of lights that allow for a perceivable representation of their state or some message. This communication is implicit and only happens when looked at.

The LOOK step allows it to see a snapshot of the positions of all robots and their configuration of lights, the COMPUTE step generates a target adjacent vertex and MOVE changes the robot's position to that target. The state of the robot encodes its local variables.

There also is the extended version of the model (EALR), where not all robots need to be present at the first moment, their appearance is asynchronous and they might crash at any moment, remaining visible. This is represented with a negative positive integer that is shown in its light. It becomes part of the scheduler's tasks to decide when each robot appears and disappears.

4.1.1.2 Asynchronous wait-free multiprocess read/write shared memory model

This model for distributed computing is composed of $N \geq 2$ asynchronous processes where at most $N - 1$ may crash and stop its execution. They have view access to a memory M and write access to a single register $M[i]$. The WRITE$_i$(x) and SNAPSHOT actions are atomic. As they are wait-free, their instructions cannot involve waiting for steps of another process.

4.1.1.3 Simulating ALR in WFSM

The ALR model can be simulated in the WSFM model considering a process q_i for each robot p_i. The register allocated for each process will store the state of p_i, its position and light value. The MOVE$_i(v_i, r_i)$ operation will be simulated via a WRITE$_i((v_i, r_i))$, and LOOK(G) will be simulated via a SNAPSHOT(M), all being atomic.

4.1.1.4 Equivalence of EALR and WFSM

The two models are presented as equivalent showing how one can simulate the other, similarly to the way presented in the previous segment, with the addition of the value $\perp$ for those robots that are not yet active. This leads to theorem 4.1.

Theorem 4.1 *A robot task T on G is solvable in the EALR model with N 2 robots tolerating f failures if and only if T is solvable in the WFSM model with N 2 processes tolerating up to f failures.*

The first direction of the equivalence is made considering a an algorithm $\mathcal{A}$ that solves the task $\mathcal{T}$, which tolerates up to f failures, in the EALR model. We can solve it in the WFSM model with N processes, tolerating up to f failures by simulating $\mathcal{A}$, where there is a process p_i simulating each robot p_i. For this purpose, the code of each robot is simulated step by step, and the MOVE and LOOK operations are done so atomically.

For the simulation in the reversed order, we consider an algorithm $\mathcal{A}$ that solves a robot task $\mathcal{T}$ with $N \geq 2$ processes, tolerating up to f failures, in the WFSM model. $\mathcal{T}$ can be solved in EALR simulating $\mathcal{A}$ the other way around, where each robot r_i simulates the code of each process q_i. For this, each robot will store the value of the respective register in the shared memory M, and won't modify it. They are also assigned to vertices v_i in the graph G, which is connected, and will not move. WRITE$_i(x)$ operations are simulated by storing that value in the robot's light with MOVE(v_i, x). The LOOK(G) and SNAPSHOT(M) are used similarly.

4.1.2 Gathering and binary consensus

4.1.2.1 Gathering with termination

The tackled problem is that of gathering, where the multiple robots attempt to decide on a common vertex. Termination is required in the problem statement, that is, all robots must be capable of deciding on a next vertex in finite time. This is particularly interesting for the asynchronous models, as the robots have arbitrarily long operations with no guaranteed bounds for the delay, and this way it is possible to know that the system will end after a bounded number of cycles. Weaker termination properties are not enough as waiting for other robots to finish before performing an operation wouldn't work.

Gathering with termination has to respect the following properties:

1. TERMINATION. Every robot decides a vertex in a bounded number of LCM cycles.
2. VALIDITY. The decided vertex cannot be fixed in advance.
3. AGREEMENT. All decided vertices are the same.

4.1.2.2 Binary consensus problem

The binary consensus problem, within distributed computing, corresponds to $N \geq 2$ processes starting with a value from $w \in \{1, 2\}$, such that their final decision respects the following properties:

1. TERMINATION. Every correct process decides a value.
2. VALIDITY. Every correct process decides a value that it is proposed by at least one process.
3. CONSENSUS-AGREEMENT. All decided values are the same.

First, we have to consider p_i-solo executions, where p_i is the only one taking steps from a configuration C until it decides, and all other robots take steps afterwards. This is important as it guarantees that a process will not be executed concurrently with others, and all p_i with the same configuration C will arrive at the same decision in the same number of steps, as they are deterministic.

Now we consider the *solo-trivial* property for a gathering algorithm, meaning that there is a vertex $\hat{u}$, which will be decided by every p_i in a p_i-solo execution starting at C. The non-existence of this property will be relevant as it means that there is at least one configuration C from which that two robots will decide on two different vertices, in a r_i-solo execution.

A proof can be made using an algorithm $\mathcal{A}$, which does not possess the solo-trivial property and satisfies TERMINATION and CONSENSUS-AGREEMENT in the ALR model, in order to create Algorithm 3, where binary consensus is solvable for two processes in WFSM model.

Algorithm 3 Binary Consensus for N=2 processes tolerating one failure. Code for process q_i

---

function BINARYCONSENSUS(i, w)
    $M[i] \leftarrow w$
    $v \leftarrow$ Decision of r_i in the simulation of $\mathcal{A}$ starting from C ▷ $\mathcal{A}$ solves gathering for two robots, so v will be the same vertice for both.
    if $v = x_1$ **then** ▷ The decision of $\mathcal{A}$ determines in which value to have consensus.
        return $M[1]$
    else
        return $M[2]$
    end if
end function

The fact that $\mathcal{A}$ solves the binary consensus problem is proved by showing that it respects the problem's properties.

1. TERMINATION. It happens as $\mathcal{A}$ satisfies Termination.
2. VALIDITY. For a different value than the proposed to be decided, the executions would have to simulate something other than a r_1-solo execution of $\mathcal{A}$ from C, which is not the case. The executions have ids corresponding to the order and they are not concurrent.
3. CONSENSUS-AGREEMENT. It will be respected as $\mathcal{A}$ satisfies the Agreement property of gathering with termination, so all decide on the same value.

4.1.2.3 Weakened gathering problems

Two weakened versions of the gathering problem are presented, both dealing with Validity properties for what it is to be sufficiently close for the gathering. They still deal with instances where termination is respected.

It is also noticeable that when dealing with only two robots, the edge-gathering and 1-gathering are the same problem. Also that solutions to the edge-gathering also solve the 1-gathering, but not the other way around.

4.1.2.4 Edge-gathering

In this instance, the gathering is relaxed so that the robots only have to share an edge, meaning that they have two options for where to decide. For this variation, it has to respect the following properties.

1. TERMINATION. Every correct robot decides as vertex in G in a bounded number of LCM cycles.
2. VALIDITY. All robots must decide on the same vertex (rep. edge) if they start on the same vertex (resp. edge).
3. EDGE-AGREEMENT. All decided vertices belong to the same edge.

4.1.2.5 1-Gathering

This instance extends the idea of edge-gathering to the idea of convergence to a maximum distance of 1, where the name comes from. Here, the robots have to decide a position at this maximum distance, which means that they should all belong to the same complete sub-graph. The following properties must be respected.

1. TERMINATION. Every correct robot decides as vertex in G in a bounded number of LCM cycles.
2. VALIDITY. All robots must decide on vertices of the same subgraph if they start inside of a complete subgraph.
3. 1-AGREEMENT. All decided vertices belong to the same complete subgraph.

4.1.3 Results

The results of the article are summarized in the following table 4.1.

Table 4.1 – Results from Alcántara et al. (2019).

Problem in EALR	Robots	Lights	Graph	Solvability
Gathering	$N \geq 2$	unbounded	Connected	Impossible: Th. 4.2
Edge-gathering	$N \geq 2$	$diam(T) - 1$	Tree	Possible: Th. 4.3
	$N \geq 3$	0	Connected with $diam(G) \geq 3$[1]	Impossible: Th. 4.4
	$N \geq 3$	unbounded	With a cycle	Impossible: Th. 4.5
1-Gathering	$N \geq 2$	$diam(T) - 1$	Tree	Possible: Th. 4.3
	$N \geq 2$	0	Dominating vertex[2]	Possible: Th. 4.6
	$N \geq 2$	bounded	The clique graph of G is a tree[3, 4]	Possible: Th. 4.7
	$N \geq 3$	0	$diam(G) \geq 3$ and no triangles	Impossible: Th. 4.4
	$N \geq 3$	unbounded	With cycles and no triangles	Impossible: Th. 4.9

4.1.3.1 Gathering

The first main result is exposed in theorem 4.2 right after the connection between the ALR/EALR and WFSM models via the Binary Consensus problem.

Theorem 4.2 *The gathering problem with termination is unsolvable by any algorithm in the ALR model, even if robots have powerful capabilities, namely, they are non-oblivious and able to detect multiplicities, share the same labeling of G and have an unbounded number of lights.*

This results comes from the fact that the same algorithm that would solve the gathering in ALR would solve the binary consensus in WFSM, which has already been proved impossible. The decision of the gathering problem requires it not to have the solo-trivial property, because they cannot have a predefined vertex for convergence per the VALIDITY property, which would happen if that property held, and without the solo-trivial it implies that the binary consensus is solvable. It is interesting to note that the impossibility comes from the validity condition, and not the capacities of the robots. It is necessary to drop one of the properties in order to solve it.

The point is to have an algorithm not respecting solo-trivial property, meaning that it can be used to solve binary consensus with an $EALR \rightarrow WFSM$ simulation.

[1] $diam(G)$ is the length of the shortest path between the vertices the most far apart.

[2] Dominating vertex is the unique vertex with the most connecting edges.

[3] A clique is a subset of the graph where every two distinct vertices are connected, that is, a subgraph that is complete.

[4] Clique graph of G is a new graph where each clique in G is reduced to a single vertex.

No gathering algorithm that respects validity will have the solo-trivial property, which would mean that they can also solve binary consensus as well. Since binary consensus is knowingly impossible to solve, any algorithm that solves gathering while respecting validity will also be impossible to produce.

4.1.3.2 Edge-gathering

Algorithm 4 is proposed for the solution of the edge-gathering problem.

Algorithm 4 Edge-Gathering algorithm for $N \geq 2$ robots on any tree $T = (V, E)$. Code for robot p_i.

function EDGE-GATHERINGTREES(v_i, T)
  Move$(v_i, 0)$
  for $r_i \leftarrow 1$ **to** $diam(t) - 1$ **do**
    $view_i \leftarrow Look(T)$
    $max\_round_i \leftarrow \max\{r_j : (*, r_j) \in view_i\}$ ▷ largest round value among the robots
    $S_i \leftarrow \{v_j : (v_j, max\_round_i) \in view_i \vee v_j = v_i\}$
    $T_i \leftarrow$ smallest subset of T spanning all vertices in S_i ▷ smallest set of edges connecting all nodes with robots in the max round value and p_i
    if v_i is a leaf of $T_i \wedge diam(T_i) > 0$ **then**
      $v_i \leftarrow$ vertex of T_i that is adjacent to v_i ▷ p_i moves deeper into the tree if in a leaf node
    end if
    Move$(v_i, 0)$
  end for
return v_i
end function

It works by making the robots in the maximal rounds dictate the destination of the others, as they are the only ones considered at each robot's execution. All robots will either be at a leaf node among the maximal round ones and move deeper inside it, or stay still, either leading to the gathering or having their node become a leaf at some point. Theorem 4.3 is demonstrated in the sequence.

Theorem 4.3 *Algorithm 4 solves the edge-gathering problem in the EALR model for $N \geq 2$ robots on any tree T in $diam(T) - 1$ rounds using $diam(T) - 1$ distinct light colors and tolerating up to $N - 1$ crash failures.*

This is proved with several lemmas that demonstrate that it holds within the VALIDITY and EDGE-GATHERING properties. This is helped with proofs that all robots will always remain inside the minimum subtree that contains their initial vertices. And with the proof that, considering that the largest number of rounds before edge-gathering is $diam(T) - 1$, the maximum distance between two robots in their rounds r and m will always be constrained by $diam(T) - min(r, m)$. With their positions limited to the initial subtree and with them always approaching, this variation of gathering is proved solvable.

The following results about the edge-gathering problem are about their impossibility in specific configurations. They are made after the equivalence is established between the EALR and WFSM models.

Theorem 4.4 *Let G be a connected graph with $diam(G) \geq 3$. Then, no algorithm $\mathcal{A}$ solves edge-gathering on G for $N \geq 3$ robots without lights in the* STRONG *version of the EALR model, with at most two robots failing.*

This first impossibility result makes the statement that lights are *necessary* for edge-gathering to be solvable. It is shown first the impossibility to solve with 3 processes tolerating up to 2 failures, and then extended to N processes using the result on the BG simulation Borowsky et al. (2001). The BG simulation helps by letting us consider an extension from $t+1$ processes and t maximum failures to N processes and also t failures. Also must be noted that the STRONG version consists of robots that are non-oblivious, non-anonymous, non-disoriented, share the same labeling of G and can detect multiplicities, meaning that the problem is not in its capacity.

It is contradictory to solve edge-gathering with $N \geq 3$ processes and up to 2 failures, as without lights it is not possible to have more information than the positions, qualifying it as restricted algorithm, and if it was possible to have a restricted algorithm $\mathcal{A}$ to solve it, then via the BG simulation there would be an algorithm $\mathcal{B}$ capable of solving it for $N = 3$ processes and up to 2 failures. VALIDITY and EDGE-AGREEMENT cannot exist at the same time as the lack of lights does not allow the robots to distinguish certain executions and, among those, VALIDITY requires the same decision to be made in cases that it would not respect EDGE-AGREEMENT, making it impossible for Algorithm $\mathcal{B}$ to exist, and consequently also Algorithm $\mathcal{A}$.

Theorem 4.5 now shows how even with an unbounded number of lights, the existence of a cycle makes it impossible to solve edge-gathering. Its proof is done by showing that if such algorithm existed, it would solve the *k-Set agreement* problem for 3 processes tolerating 2 failures, which is impossible in the WFSM model when $k = 2$. The *k-Set agreement* problem extends the *Binary consensus* problem by accepting up to *k* different values to be decided.

Theorem 4.5 *Let G be a connected cyclic graph, then there is no algorithm that solves edge-gathering on G for $N \geq 3$ robots tolerating two failures in the* STRONG *version of the EALR model.*

First, the vertices are remapped into a simple cycle with three distinct and consecutive vertices, so that v_2 is mapped to 2, v_3 is mapped to 3 and all the others to 1, and it is assumed the existence of an algorithm $\mathcal{A}$ that solved the edge-gathering with 3 robots in a graph with a cycle. It is considered that there is an algorithm that solves edge-gathering for 2 robots in a path P without lights and, equivalently, an algorithm $\mathcal{B}$ that solves edge-gathering on P in WFSM for two processes q_1 and q_2.

Algorithm 5 uses $\mathcal{A}$ and $\mathcal{B}$ to solve the *2-Set Agreement* problem for 3 processes with up to 2 failures.

Algorithm 5 2-Set Agreement algorithm for $N = 3$ processes tolerating 2 failures. Code for process q_i.

function SETAGREEMENT(v)
  $M[i] \leftarrow v_i$
  if $i = 3$ **then**
    $u \leftarrow v_3$ ▷ q_3 is fixed with v_3 for algorithm $\mathcal{A}$
  else
    $u \leftarrow \mathcal{B}$.**EdgeGathering2Robots**(v_i, P) ▷ $\mathcal{A}$ will receive at most two adjacent nodes for the initial position of q_1 and q_2
  end if
  $y_j \leftarrow \mathcal{A}$.**EdgeGatheringCycle**($u, G$)
  if $y_j \in \{v_1, v_2, v_3\}$ **then**
    return $M[j]$
  end if
    return $M[1]$
end function

It works by mapping each of the 3 vertices to the corresponding index in the memory M, so that the process p_3 can be ignored and edge-gathering is solved with the remaining two using $\mathcal{B}$. Because of how the vertices are arranged, the now determined input to $\mathcal{A}$ will always decide on vertices that share an edge, assuming that $\mathcal{B}$ works as intended. This result is impossible, as it has already been proved for the *2-Set Agreement* problem.

This result for $N = 3$ (after showing that VALIDITY and EDGE-AGREEMENT properties are respected) is extended to $N \geq 3$ and 2 failures again using the BG *simulation*, concluding the proof with the equivalency between WFSM and EALR models.

4.1.3.3 1-gathering

The first result for 1-gathering comes with theorem 4.3 as well, as an edge-gathering constitutes an instance of gathering on a complete subgraph.

It is possible, however, to achieve 1-gathering in relaxed configurations, that need not respect the tree structure. This is first shown for graphs with a dominating vertex and then for graphs with a clique graph that has a tree structure, both of them for $N \geq 2$ robots.

Theorem 4.6 *Let G be a graph with at least one dominating vertex, then, 1-gathering is solvable in the EALR model for $N \geq 2$ robots without lights on G in only one round and tolerating up to $N-1$ crash failures.*

Theorem 4.6 is supported by algorithm 6, which solves in a single round the gathering. It either stays in the same vertex it started if all perceivable vertices belong to the same complete graph, otherwise it moves to any dominating vertex. As a dominating vertex will be at most at distance 1 of all other vertices, it is impossible to have a distance of 2 or larger. This respects the 1-AGREEMENT and VALIDITY properties, as they will either not execute a MOVE operation if already satisfied or solve it in a single operation. Note that lights are not needed for this.

Algorithm 6 1-gathering algorithm for $N \geq 2$ robots on any connected graph $G = (V, E)$ with at least one dominating vertex. Code for robot p_i.

function 1GATHERINGDOMINATINGVERTEX(v_i, G)
  Move$(v_i, 0)$
  $view_i \leftarrow$ **Look**(G)
  if $\exists v_j \in view_i$ such that $dist(v_i, v_j) \geq 2$ **then** ▷ It is not in a complete subgraph.
    $v_i \leftarrow \hat{v}$ ▷ Target the dominant vertex $\hat{v}$.
    Move$(v_i, 0)$
  end if
  return v_i
end function

Theorem 4.7 *Let G be a graph such that $K(G)$ is a tree. Then, 1-gathering is solvable in EALR for $N \geq 2$ robots on G in at most $(2 \cdot diam(K(G)) + diam(G))$ rounds, using at most $4 \cdot diam(K(G)) \cdot |V(G)| \cdot |V(K(G))|$ light colors and tolerating up to $N-1$ crash failures.*

The other result in a relaxed graph structure shown in 4.7. This proof is done by reducing the problem to a distributed version of the same solved by 4. For this, the notion of clique graphs $K(G)$ are used, where each vertex represents a maximal complete subgraph of G, with the edges representing that those two subgraphs share vertices (their intersection is not empty). It is also used a graph $K'(G)$, a subdivision of $K(G)$ where every edge is replaced by a pair that introduces a new vertex between them, meaning that every $\{K_x, K_y\} \in E(K(G))$ is replaced with $\{K_x, K_{xy}\}$ and $\{K_{xy}, K_y\}$.

For this reduction, a pair of functions $f_{in} : V(G) \mapsto V(K(G))$ and $f_{out} : V(K'(G)) \mapsto V(G)$ is needed, where f_{in} transforms inputs to 1-gathering on G to inputs to the Edge-gathering algorithm, and $_{out}$ transforms outputs from it to the ones from 1-gathering on G. The lights also gain extra functionality, where they display the input vertex v_i, the vertex where p_i stands on the simulated graph $K'(G)$, and the round number on the simulated Algorithm 4.

This is all used in Algorithm 7. Note that a robot may move while another is still deciding in the simulation of Algorithm 4 from f_{in}, for that purpose it preserves its initial position in the lights, so that information stays accessible independent of their actual position. This means that their actual position will not be used for computations, as we can see in line 5.

Algorithm 7 1-gathering algorithm for $N \geq 2$ robots on any connected graph $G = (V, E)$ whose clique graph K(G) is a tree. Code for robot p_i.

```
 1: function 1GATHERINGCLIQUETREE(v_i, G, f_in, f_out)
 2:     Move(v_i, v_i|⊥|⊥)
 3:     u_i ← SimulateEdgeGatheringTrees(f_in(v_i), K'(G))   ▷ it doesn't matter where
        they are inside a complete graph, so all positions are treated the same
 4:     r_i ← light value of p_i   ▷ the light has to be preserved, as it may move, but the
        initial value will be used by others
 5:     S_i ← {w_i : (*, w_i| * |*) ∈ Look(G)}   ▷ the initial position of all robots is
        obtained from the lights, as they will preserve that information even while moving
 6:     if exists an initial position w_i ∈ S_i such that w_i belongs to the complete subgraph
        u_i, i.e. w_i ∈ V(u_i) then
 7:         x_i ← w_i   ▷ it moves to a complete subgraph if an initial positions requires for
        validity
 8:     else
 9:         x_i ← f_out(u_i)   ▷ moves to the output of Algo 4
10:     end if
11:     while x_i ≠ v_i do   ▷ loop to reach the target vertex
12:         v_i ← closes vertex to x_i
13:         Move(v_i, r_i)
14:     end while
15: return v_i
16: end function
```

Theorem 4.8 *Let G be a connected graph with $diam(G) \geq 3$ and no triangles. Then, there is no algorithm $\mathcal{A}$ that solves 1-gathering problem on G for $N \geq 3$ robots without lights in the Strong version of the EALR model, with at most two robots failing.*

Now again we have more impossibility results. Theorem 4.8 on 1-gathering also comes from Theorem 4.4, being the same situation when there are no triangles, which makes edge-gathering the only possibility at distance 1.

Theorem 4.9 *Let G be a connected cyclic graph without triangles. There is no algorithm that solves 1-gathering on G for $N \geq 3$ robots tolerating two failures in the* STRONG *version of the EALR model.*

Similarly as it is done for theorem 4.8, Theorem 4.9 extends from the idea of Theorem 4.5 with the fact that 1-gathering is equivalent to edge-gathering when there are no triangles, being possible to hold distance 1 only in the vertices that share an edge.

4.1.4 Robot tasks

It is possible to represent initial and final positions of robots in a graph G via combinatorial topology constructs called simplexes, where multiple of them can be bundled into complexes that represent all configurations allowed in a given *robot task*.

Definition 4.1 *A* robot task *$\mathcal{T}$ on G is a triple $\langle \mathcal{I}, \mathcal{O}, \triangle \rangle$, where:*

1. *$\mathcal{I}$ is an input complex, and is composed of all input simplexes. An input simplex σ represents a possible combination of initial position for all robots.*
2. *$\mathcal{O}$ is an output complex, and is composed of all output simplexes. An output simplex τ represents a possible combination of vertices where the robots are allowed to end, a set of decided vertices.*
3. *$\triangle$ is a monotonic carrier map between the input and output complexes.*

Both $\mathcal{I}$ and $\mathcal{O}$ are closed under containment, meaning that any subset of an input (resp. output) simplex will also be an input (resp. output) simplex. In other words, robots are allowed to start (resp. end) in any subset of initial (resp. final) vertices.

This formalization allows for the representation of tasks such as gathering. In this case, AGREEMENT property is held by the output simplexes in $\mathcal{O}$ being composed of all sets with a single vertex of G, and the VALIDITY property by forcing that all input simplexes with cardinality 1 will also be their output via $\triangle$ ($|\sigma| = 1, \triangle(\sigma) = \{\sigma\}$, otherwise $\triangle(\sigma) = \mathcal{O}$).

In the case of edge-gathering, the output complex $\mathcal{O}$ of the previous definition has added the sets of two vertices that belong to the same edge in G, in order to respect EDGE-AGREEMENT, and the VALIDITY constraint it updated so that two robots sharing an edge can choose either or both the vertices ($\triangle(\sigma) = \{\sigma, \{v_1\}, \{v_2\}\}$)

Originally it was defended that not all problems could be represented as robot tasks, as of all tasks are defined in terms of the vertices of G, and all positions in the output complex will be valid alongside their subsets. A task definition also cannot be specified so that certain robots end up or not in specific vertices. An example is the task of forming a connected graph, which should not be representable, as the final positions allow for any number of robots to be placed, with no way to constrain the placement of the vertices around.

Note that while this framework supports the case of failure of robots, it does not allow take into consideration the limits in the sensors of the robots, considering always unrestricted communication and visibility of the map. A new understanding, with a more realistic view, will be discussed in chapter 6.

4.1.4.1 Solvable robot tasks

A topological characterization is proposed of solvable robot tasks following an approach for distributed computing. It is a consequence if the equivalence between the EALR and WFSM models and a theorem from Herlihy and Shavit (1999) stating that a robot task is solvable in WFSM if and only if there is a subdivision of $\mathcal{I}$ and a simplicial decomposition map to $\mathcal{O}$ respecting $\triangle$. Theorem 4.10 is then proposed.

Theorem 4.10 *A robot task on G, $\langle \mathcal{I}, \mathcal{O}, \triangle \rangle$ is solvable for N robots in the EALR model tolerating $N-1$ failures if and only if there is a subdivision $Subd(\mathcal{I})$ and a simplicial map δ from $Subd(\mathcal{I})$ to $\mathcal{O}$, such that for every input simplex σ, $\delta(Subd(\sigma)) \subseteq \triangle(\sigma)$.*

A task $\mathcal{T}$ makes use of processes IDs, which translates to colored vertices in the input and output simplexes, corresponding to those IDs. $\mathcal{T}$ will have a wait-free read/write protocol if and only if its input complex $\mathcal{I}$ has a chromatic subdivision X (note that it is colored from the IDs), and this subdivision can be mapped to the output complex $\mathcal{O}$ in a color-preserving manner. Such map $\delta : X(\mathcal{I}) \mapsto \mathcal{O}$ must be done so that for each simplex $s' \in X(s), \delta(s')$ is carried by $\triangle(s)$, in other words, each simplex obtained from the subdivision of another simplex, must have its mapping to the output simplex via δ carried by its mapping via $\triangle$.

The chromatic subdivision $X(\mathcal{I})$ is the final state of a distributed computation after an execution that starts with $\mathcal{I}$. Locally, if we consider s, a simplex representing a single initial configuration in $\mathcal{I}$, $X(s)$ will give us the final state of all processes that started as s after being subdivided by a distributed algorithm, while $V(X(s))$ will be the final state of each process. This final states obtained from $Subd(\mathcal{I})$ are then mapped to their corresponding decisions in the complex of possible decisions $\mathcal{O}$ via the decision map δ. This is visible in Figure 4.1.

Remark 4.1 *The map $\triangle$ lets us obtain the decisions made from the combination of initial positions in $\mathcal{I}$, among all possible decisions in $\mathcal{O}$. This represents the properties that must be respected for each problem. The color-preserving simplicial mapping δ lets us do the same after the execution of a distributed algorithm, represented by $X(\mathcal{I})$. δ must exist for a task $\mathcal{T}$ to be solvable via X.*

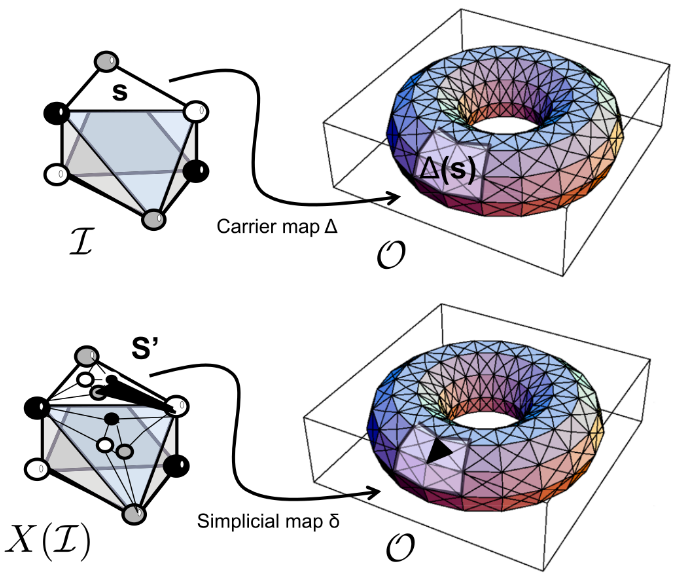


Figure 4.1 – Subdivision of $\mathcal{T}$ and mapping via δ shown in Herlihy, Kozlov and Rajsbaum (2014).

Note that while the general case is dealt with chromatic simplexes, this work has only presented the usage of colorless tasks for robots, where the decision of specific robots is not important, only the respect of the final configurations by the set of agents.

This entire comprehension of robot tasks has been reviewed, and will also be presented in chapter 6.

4.2 Myopic exploration as a robot task

Besides the study of gathering problems in the previous parts of this work, it is also interesting the case of robot exploration. This was mostly motivated by the statement in Alcántara et al. (2019) that only converging tasks could be represented as shown in section 4.1, while diverging tasks such as exploration would be impossible in the topological structure of the configuration. This has been shown wrong, making use of a different interpretation of the simplicials seen before.

As an attempt to approximate the guaranteed theoretical methods and the applicable results required by roboticists, it was also a motivation to study those problems in more realistic scenarios. One characteristic that was found relevant is the range of the robot's sensors, which has been approached in the literature mostly regarding the visibility radius when their communication depends on it. This scenario is similar to underwater robots that cannot use global communication systems due to their submersion. Here, the taxonomy, recent results and some considerations will be exposed.

4.2.1 Introduction

The considered robots in the LOOK-COMPUTE-MOVE model have the capacity of sensing their local vicinity. They are assumed to be anonymous and oblivious, with no direct means of communication. Two main variations studied:

- *Terminating exploration.* TE. each vertex is visited by at least one robot and, eventually, all robots stop. Interesting finite tasks, such as find out the boundaries of an unknown area.
- *Exclusive perpetual exploration.* EPE. each robot visits every vertex infinitely often and no two robots traverse the same edge at the same time nor visit the same vertex at the same time. Interesting for maintenance scenarios with limited space the prevent crossing or occupying the same space. In this case, no starting position has towers (multiple robots in the same vertex).

The LOOK step changes the most, as it rules the behavior of the *snapshots*. Since their sensing capabilities are limited, there is a *visibility radius* ρ, where a robot sees the rooted subgraph induced by the vertices at distance at most ρ from the itself, perceiving the number of robots and whatever information they display inside that subgraph.

The problems now have the added difficulty of reconstructing information from fewer moments when it can be gathered about the other agents. Solutions will require more steps as the MOVE operations are chosen to increase the chance of encounter when needed. The accuracy of this information changes according to the scenario, where, for example, the parameter *multiplicity detection* dictates if they can identify exactly how many robots exist or simply "more than one" on a vertex.

Other usual variations of the problem are described in Bonnet and Defago (2011):

- *Foraging*. Similar to *terminating exploration*, but the goal is replaced by finding all resources located in unknown locations.
- *Surveillance*. Equivalent to *perpetual exploration*, without the exclusivity. Usually focuses on lowering the number of robots.
- *Patrolling*. Similar to *surveillance*, but restrictions are imposed on the movement pattern so to counter a malicious agent.
- *Pursuit/Chase*. Similar to *foraging*, where the searched resources (the intruders) are not static.

All of them may change over the number of agents and constraints based on the scenario.

While the robots are usually presented as oblivious, i.e. they cannot remember more than one action in the past, they are capable of some communication in the form of lights or a local whiteboard. They have their visibility limited by a parameter ρ and are only perceived in vertices. They are not always equipped with lights, but it may be needed if we chose to reduce it to a version of gathering, as Theorems 4.4 and 4.9 show that without lights the number of solvable families is severely reduced. They are considered sometimes considered *opaque*, limiting the view of other robots if aligned.

Most of the works start from a general finite connected graph, and they restrict to specific families in order to solve, such as in Flocchini, Prencipe and Santoro (2019), Bramas, Lafourcade and Devismes (2021). The most common families are Rings, Tree, Grids and Tori, but there are also efforts for more general impossibility conclusions. Time-varying graphs (TVA) are also considered in Gotoh et al. (2021), where a temporal domain is considered together with a edge presence function, describing in which time steps the edge exists.

4.2.2 Known results

Usually the focus is on the smallest or largest number of robots, $\kappa^{-}(n)$ and $\kappa^{+}(n)$, of any n-vertex graph of a given family, discussed in Flocchini, Prencipe and Santoro (2019). Tables 4.2 and 4.2.2 present an extract of the results found in the literature.

While considering only the case of myopic robots, we have the following table with the latest results:

Table 4.2 – Recent results for myopic exploration tasks.

Problem	Graph	Synchrony	Visibility	Colors	Result
TE	grid	Fsync	2	2	Possible, 2 robots, CC. [12]
TE	grid	Fsync	2	2	Possible, 2 robots. [12]
TE	grid	Fsync	2	1	Possible, 3 robots, CC. [12]
TE	grid	Fsync	2	1	Possible, 3 robots. [12]
TE	grid	Fsync	1	3	Possible, 2 robots, CC. [12]
TE	grid	Fsync	1	3	Possible, 2 robots. [12]
TE	grid	Fsync	1	2	Possible, 3 robots, CC. [12]
TE	grid	Fsync	1	2	Possible, 3 robots.[12]
TE	grid	Ssync, Async	2	3	Possible, 2 robots, CC.[12]
TE	grid	Ssync, Async	2	3	Possible, 2 robots.[12]
TE	grid	Ssync, Async	2	2	Possible, 2 robots, CC.[12]
TE	grid	Ssync, Async	2	2	Possible, 2 robots.[12]
TE	grid	Ssync, Async	1	3	Possible, 3 robots, CC. [12]
TE	grid	Ssync, Async	1	3	Possible, 3 robots.[12]
TE	ring	FSync	1	1	Possible, 5 robots. [13]
TE	ring	SSync, Async	1	1	Impossible. [13]
TE	ring	Fsymc, Ssync, Async	≥ 2	≥ 2	Possible, 3 robots. [14]
TE	ring	Fsync	1	≥ 2	Possible, 3 robots. [15]
TE	ring	Ssync, Async	1	≥ 2	Possible, 4 robots. [15]
PE	ring	Fsync	1	≥ 2	Possible, 2 robots.[15]
PE	ring	Ssync, Async	1	≥ 2	Possible, 3 robots. [15]
PE	ring	Fsymc, Ssync, Async	≥ 2	≥ 2	Possible, 2 robots. [16]

[l]in Bramas, Lafourcade and Devismes (2021)
[m]in Datta et al. (2013b)
[n]in Nagahama, Ooshita and Inoue (2019)
[o]in Ooshita and Tixeuil (2021)
[p]in Nagahama, Ooshita and Inoue (2019)

From these references, it is possible to notice some main takeaways. First, the breakage of symmetries is a key part of the problem to be addressed in order to render it solvable, such that it is often the case of few robots that perform the exploration and others are used to keep track of the process while staying off symmetric configurations. The termination is based on specific configurations of the robots, such as positions of the robots in the map, sometimes making use of the communications channels, with lights or whiteboards. Lastly, that common chirality seems to be an efficient property in reducing

Problem	Graph	Sync	# Robots	# Lights	Visibility	Labels	Result
EPE	Grid	Fsync	1	finite	finite	local	Impossible. Th 3.2 [9]
EPE	Grid	Fsync	2	1	finite	local	Impossible. Th 3.4 [9]
EPE	Grid	Fsync	2	2	1	local	Impossible. Th 3.5 [9]
EPE	Grid	Fsync	3	1	1	local	Impossible. Th 3.6 [9]
EPE	Grid	Fsync	2	3	1	local	Possible. Alg $Vone_3^2$ [9]
EPE	Grid	Fsync	2	2	2	local	Possible. Alg $Vtwo_2^2$ [9]
EPE	Grid	Fsync	3	1	2	local	Possible. Alg $Vtwo_1^3$ [9]
PE	TVG[5], $\mathcal{H}$[6]	Fsync,Ssync	$k \geq 2\eta + 1$[7]	0	whiteboard	local	Possible. Th. 6 [6]
PE	TVG, $\mathcal{W}(l)$[8]	Ssync	$k \geq 2l + 1$	0	whiteboard	local	Possible. Th. 10 [10]
PE	TVG, $\mathcal{W}(l)$	Ssync	$k \geq 2l$	0	whiteboard	local+leader	Possible. Th. 23[10]
PE	TVG, $\mathcal{W}(l)$	Fsync	$k \geq 2l$	0	whiteboard	local	Possible. Th. 19[10]
PE	TVG, $\mathcal{W}(l)$	Fsync	$k > 2l - 1$	0	whiteboard	local+leader	Possible. Th. 34[10]
TE,EPE	n-Ring, $n \geq 3$	Fsync	$k < n/2$	0	global	global	Impossible. Lm 9.1 [k]

Figure 4.2 – Results in the literature for exploration tasks in LCM.

[i] in Bramas, Lafourcade and Devismes (2021)
[j] in Gotoh et al. (2021)
[k] in Flocchini, Prencipe and Santoro (2019)
[e] Time-Varying Graph, with unknown topology.
[f] Temporally connected. The graph is always connected over the time domain.
[g] η is the evanescence of the graph, the number of transient edges.
[h] l-bounded 1-interval. All graphs are connected with at most l transient edges, those that are not always present.

the difficulty of the problem, and is a reasonable assumption when understood as the existence of a compass.

4.2.3 Model for exploration

From this bibliography research, a model has been created for the future developments, aggregating the factors that were found the most interesting to be studied in a first moment. Scenarios with underwater robots are envisioned, as simple problems are still complicated to deal with, given the heavily constrained information available.

4.2.3.1 Proposed robot model

- Asynchronous, as no common scheduler is available with global communication;
- Luminous, with unbounded lights. This models the implicit communication that has to be observed in order to succed.
- Myopic, with visibility limited to $\rho = 1$.

In order to emulate a scenario with underwater drones, communication and visibility have to be restricted. Considering that any graph family will be a discretization of the aquatic environment, we can associate the sensing limits to that discretization. It is possible to give the interpretation that the current node is a zone with guaranteed visibility, and the adjacent nodes are the ones at the limit of that, where sensing becomes likely but not guaranteed. This fits the usage of grid graphs, describing the limitations in each of the physical dimensions. In the case of terminating exploration, the problem statement implies the existence of a configuration where all robots decide not to move. This configuration must not be a part of the initial configurations.

4.2.3.2 Simulating myopic EALR in WFSM

In order to simulate EALR in WFSM, the MOVE(v_j, c_j) operations of robot r_i are replaced by WRITE$((v_j, r_j))$ of process p_i, where the initial data stored at the register is always the current vertex of r_i, and this information is used for the simulated COMPUTE operation. The same operation can be adapted in order to ignore values from registers that start with vertices deemed to far away from the current process'. Similarly, the SNAPSHOT operation can be adapted to only return the contents of registers with nearby vertices expressed at the beginning of the data, simulating the restricted visibility of the LOOK step.

The converse simulation can make use of the proof from Alcántara et al. (2019), where instead of robots assuming any position in a connected graph, they assume positions that respect the visibility limit of the model. This still has to be developed, and an attempt is seen bellow.

4.2.3.3 First idea for EALR

This may be representable inside the *robot tasks* if we consider that they have to end in a distinguishable vertex, such as the dominant one. The point is that they are only allowed to go to that vertex once they have explored all open edges. Some other family of graphs would be more interesting though.

Usage of lights to store the history of chosen paths. If we assume common sense of direction, i.e. any two robots a, b with a numbering function for the edges, leaving edges $\lambda(V_i)$, will have $\lambda_a(V_i) = \lambda_b(V_i)$, for any vertex $V_i \in G$.

The lights store a history of pairs with the chosen port and the degree of the vertex in which that choice was made, such as $L = \{(P_1, D_1), (P_1, D_1), \cdots, (P_N, D_N)\}$. An alternative is also a sort of heap tree where we leave “empty” spaces for the unexplored edges. In either case that structure will be completed as they meet other robots, and fill the map. It may be better to use lights to signal the end of all explorations, and terminate when consensus is reached about it.

And the following operations are considered:

- LOOK. They have a SNAPSHOT of the vertices that are in path with distance at most ρ.
- COMPUTE. At each time they follow a path to the closest unexplored edge. If there are no unexplored edges, they go to the dominant vertex and terminate. They may cross the dominant edge before the final path.
- MOVE. They move to the next vertex in the computed path.

4.3 Simple robot task in line graph

As part of the study on how the simplicial complexes work for a proper robot task formalization, their usage to represent configurations instead of simply positions in the graph was considered. The configuration interpretation includes everything in the state of the robots, such as their lights, their position and, if needed be, their history of positions. As a tool for verifying solvability, all information is accessible for the analysis.

This approach was studied with the case of a robot moving in a line graph, composed of three vertices and two edges connecting them. The interest is to expand it to the case of grid graphs, where each vertex that is not a boundary has 4 neighbors, two in each dimension. Grid graphs were considered in section 4.2.3. The 1-gathering problem is shown in the example, where robots have to find themselves with at most 1 edge separating them all, as this is the simplest problem in the line graph that is solvable after one step, while having the input and output complexes different.

Each robot has three possible starting positions and can either move to the left or right in the representation in the top of image 4.3, next to the input complex $\mathcal{I}$ and output complex $\mathcal{O}$, where the configurations consist only of their positions. in the case of $\mathcal{O}$, this is a shorthand for the idea that any configuration ending in those positions suffices, as one would note that more information is stored in the protocol complex. Note that the only difference is that while the two robots can start in any pair of positions in the graph, they can only end in those that share an edge, eliminating the pairs with only outsider vertices.

The four possible subdivisions are represented in image 4.3. Those consist of the cases where two robots start at the same vertex, with possibility of moving in both directions or not, and when they do not start in the same vertex, where only one has multiple movement options. All possible movements in the complete version of the protocol complex $\surd$ are assembled as in image 4.4. Note that we have two pairs of pyramids connected by the actions of staying in the same place, and that the assembly can only be done with vertices representing the same history of positions, i.e. both encode robot A starting in vertex v_1 and moving to v_2, istead of simply their current position.

Further work still will be done to expand this into a grid, but patterns are already noticeable as each pair of pyramids represents set of movements in one dimension, and they are connected by the configurations with “staying still” actions.

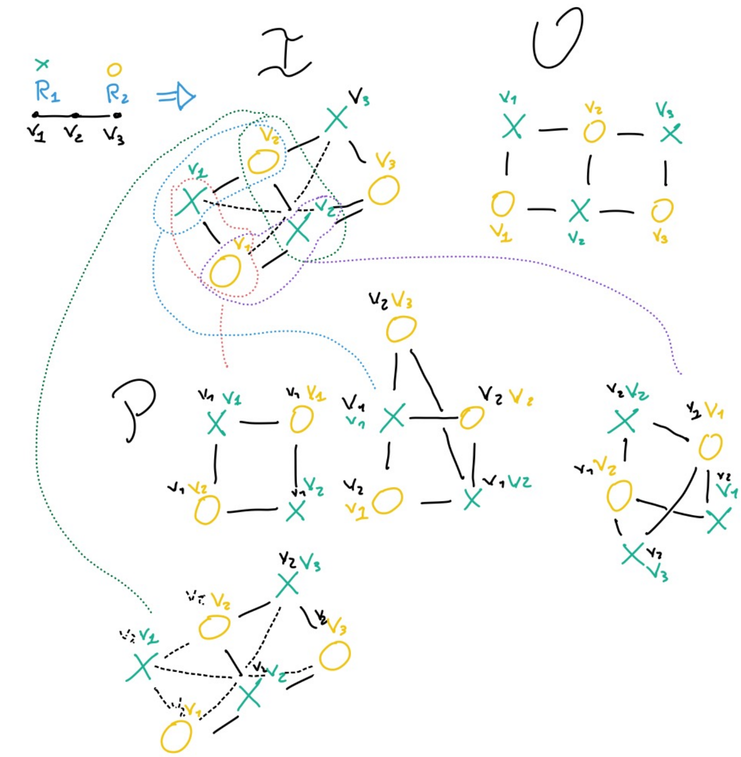


Figure 4.3 – 4 main blocks that appear as the input complex is subdivided.

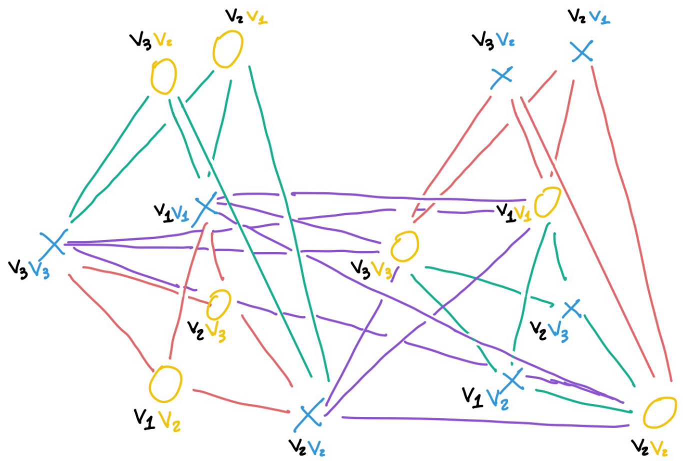


Figure 4.4 – Possible assembly of all subcomplexes related to the movement of the robot on the line.

5 FORMALIZATION OF TASKS

5.1 Robot tasks

Robot tasks may be formalized similarly to the way presented for general tasks in section 2.1.4, making use of the equivalences presented in section 4.1.

We will chose to use as base definitions 2.8 and 2.9, as the presence of exclusively functional relations is easier to abstract, which is not the case of carrier maps in 2.5 and 2.6. Note as well that these definition do not allow for the consideration of the sensing limitations of the robots, and a direct usage inspired by the robot tasks in section 4.1.4 would only for cases of unlimited visibility, which hardly corresponds with realistic scenarios.

From this, we have two possible interpretations of the protocol specification and the solvability decision map. In both cases, the existence of lights will simply increase the combinatorial complexity, either during the evolution of the movements, or in the decision map displaying the solvability.

Recall the two main structures needed for the evaluation of a task: the task definition and the specification protocol. The task definition is a triple $(\mathcal{I}, \mathcal{O}, t)$, where we represent all possible initial and final configurations in the chromatic simplicial complexes $\mathcal{I}$ and $\mathcal{O}$, while the valid pairs of input/output are filtered via the restriction function $t : \mathcal{I} \times \mathcal{O} \to \mathcal{T}$. The specification is a pair $(P, \mathcal{S})$, where P describes the subdivisions applied to $\mathcal{I}$ over the rounds, while $\mathcal{S}$ specifies how a folding between rounds must be done in order to obtain the original complex.

In the first interpretation, we consider that the specification $\mathcal{S}$ models communication, how the robots exchange information, and the decision map δ models movement. The specification is equivalent to the LOOK steps, while the decision map δ represents a reconstruction of the mission through the equivalent MOVE steps. Again, the COMPUTE step is considered to be atomically executed in sequence of a LOOK operation, without loss of generality. Solvability is a matter of finding a decision map $\delta : \mathcal{T} \to P(\mathcal{I})$ that shows the existence of a sequence of movements for all robots, given the possible communications from $P(\mathcal{I})$, that respect the visibility of the robots. While the visibility may be unlimited or not, this takes place in the communication rounds, where the ones beyond a visibility threshold are eliminated.

Alternatively, in a way that may be perceived as more natural, the specification of the robot represents how it can interact with the environment, with all possible MOVE operations encoded in the protocol complex $P(\mathcal{I})$, together with the lights as part of the state, in case they exist. The decision map δ required for a solvable task will have to map from a subset of the protocol complex $\mathcal{Q} \subseteq \mathcal{P}(\mathcal{I})$ that possesses at least one execution with the initial/terminal values specified in $\mathcal{T} \subseteq \mathcal{I} \times \mathcal{O}$, encoding the sequence of moves that need to happen in order to the attainable configurations in $\mathcal{Q}$ to be feasible. The configurations in $\mathcal{Q}$ are the accumulated history of the robots, with all vertices they have gone through, such as presented in section 4.3, with a full evolution after one step.

5.2 Formalization of robot tasks

The chosen formalization makes use of chromatic complexes, as seen in 2.1.4, instead of colorless ones. This is done in order to have the power to specify configurations for individual robots. Note that the case where all robots can assume whatever configuration acceptable, usually expressed in colorless tasks, can also be written as a general task, being simply more verbose. As the first modification, we decorate the configurations, which originally represented only the positions of the robots in the graph, with all knowledge of their state. This includes lights, used for the formalization of communication, past positions and whatever other information may be available. A robot task definition is formalized as follows.

Definition 5.1 *A robot task is a triple $(\mathcal{I}, \mathcal{O}, t)$, where:*

- *$\mathcal{I}$ is a chromatic simplicial complex, it encodes all feasible initial configurations in the task;*
- *$\mathcal{O}$ is a chromatic simplicial complex, it encodes all final configurations that satisfy the objective of the task;*
- *$t : \mathcal{T} \to \mathcal{I} \times \mathcal{O}$ is a chromatic simplicial map, it restricts which pairs of input/output configurations that are compatible with the task.*

Consider the vertex-gathering example for two robots in a graph composed of two vertices and one edge. Image 5.1 depicts the elements of definition 5.1. While they are allowed to start in any of the two vertices, as shown in $\mathcal{I}$, they must end in the same vertex, either of the two available, shown in $\mathcal{O}$. The product $\mathcal{I} \times \mathcal{O}$ generates all possible combinations of initial and final configurations, however not all of them respect validity, in which an initial configuration where the problem is already solved must not be changed by the algorithm, and those are filtered out in $\mathcal{T}$ via t.

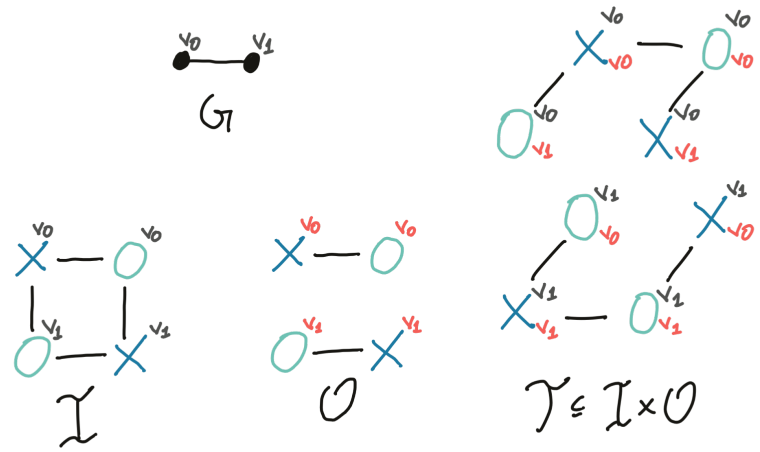


Figure 5.1 – Input complex $\mathcal{I}$, output complex $\mathcal{O}$, task complex $\mathcal{T}$ and the related morphisms in the gathering task for two robots on a graph G. Input vertex is displayed in grey and output vertex in red. No extra communication data is used.

The second part needed is a way to express the workings of the robotic system, in order to check whether it respects the criteria for solvability. This is done in the following robot specification.

Definition 5.2 *A robot specification is a triple* $(P, \mathcal{S})$*, where:*

- P *is an endofunctor operating on simplicial complexes, it generates the possible states each configuration will assume in the execution of another set of given states.*
- $\mathcal{S} : \mathcal{P} \to id$ *is a natural transformation, it encodes the specification of how the robots are capable of moving. It describes the relation between states.*

The chromatic simplicial complex $P(\mathcal{I})$ will be a combination of all paths robots can take, while respecting their specification. Each vertex will represent a series of states a robot will have gone through. Note that multiple instances of the same robot may end up at the same state after a round in the protocol complex, but each will have had a different path to it. The existence of those options in paths is the first motivation for the following solvability criteria.

Remark 5.1 *Recall that the original definition in section 2.1.4 requires the protocol carrier map to be strict, i.e.* $\Xi(\sigma \cap \tau) = \Xi(\sigma) \cap \Xi(\tau)$*. The chromatic simplicial map that models it in* $P(\mathcal{I})$ *preserves this property as it maps the entire chromatic simplicial complex into another one, preserving labeling due to its chromatic nature and carrying simplicies to simplices by definition.*

Definition 5.3 *The solvability of a robot task* $(\mathcal{I}, \mathcal{O}, t)$ *with a specification* $(P, \mathcal{S})$ *is attested by the existence of two morphisms* s *and* δ *so that diagram 5.2 commutes, where:*

- $s : \mathcal{Q} \to P(\mathcal{I})$ *is a chromatic simplicial map, acting as a restriction map that selects in* $\mathcal{Q}$ *at least one configuration from* $P(\mathcal{I})$ *that possess the same initial and final states required by the task definition* $\mathcal{T}$*. This is done in order to respect the next morphism.*
- $\delta : \mathcal{Q} \to \mathcal{T}$ *is a chromatic simplicial map that applies the restrictions of the robots' sensing capacities. In the case of unlimited vision, it will be the identity map.*

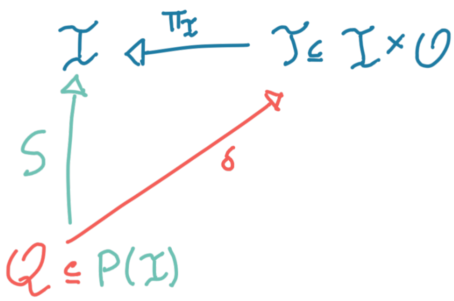


Figure 5.2 – Diagram must commute for solvability. Robot task definition in blue, specification in green and solvability criteria in red.

This formalization lets us consider multiple aspects of a mission with multiple robots. First, faulty robots that may stop working are represented by definition in lower dimensions of the simplices, as the combinatorial structures are closed under containment. The usage of inpure simplicial complexes, that is, with maximal simplices that are not of the same dimensions, we can represent configurations where robots have crashed and are no longer responsive. The case of "freeze" crashes is also representable, as it consists of robots no longer changing their states, but being observed the same way of one that takes a lot of time to update.

The δ map, as a representation of the sensing capacities of the robots, allows to filter state histories in $P(\mathcal{I})$ do not respect those limitations. The example of limited visibility works by filtering all MOVE operations that are consequent of sensing robots that could not have been seen. The intuition is to reconstruct the execution of an algorithm in configuration of state histories, while respecting the actions they could have performed.

Another problem often present in robotic implementations is the lack of precision in the robot's movements and other interactions with the environment, which can also be encoded in the specification and generate states for all posibilities. Since the simplicial complexes simply connect indistinguishable states, we have access to a natural way to represent uncertainty in these discrete environments.

6 A CONCLUSION

6.1 Developments

The developed work has addressed the initial connections presented between modal logics, combinatorial topology and the look-compute-move model for robotics. This has been motivated by the lack of practical results in guaranteed methods that are made available for robotic implementations, as well as instigating possibility of connecting different domain of research in order to share results between them.

Roboticists in search of guaranteed methods often face an entry barrier, as results are expressed mathematically in terms that usually do not intersect that what is required for robotic applications. Furthermore, it is not rare for guaranteed methods to be developed without tangible robotic implementations in mind, where the tackled problems are either too simplistic or without a clear path for adaptation into an actual algorithm applied in a robot. While the value in a gradual building of the theory for greater results is not being questioned, the proven properties are frequently lost during an adapted implementation due to the lack of ease in its use and incomplete understanding.

This has been observed to happen more often with approaches that deal with discrete variations of problems, as opposed to those in some Euclidean space. As seen in this work, the discrete approach for robotic problems offers interesting equivalence at different levels with other domains in mathematics. This motivates the efforts for its use as a more accessible expression of guaranteed methods, where the relation between the mathematical structures and consequences in a robotic mission are made clearer. An initial attempt of this was presented in chapter 5.

While a brief theoretical background is offered in chapter 2, the initial effort in understanding the aforementioned mathematical links is presented in chapter 3. The knowledge gain theorem is presented as a result in logic that can be useful when seen through the computing perspective. Finally, a concrete use of the equivalence in Armenta-Segura, Rajsbaum and Ledent (2020) is presented with the representation of the approximate agreement problem in the point of view of dynamic epistemic logic. The concept of equivalence of categories is presented in details in the appendix, alongside the categorical representation of modal logic through Kripke frames and models and distributed computing with simplicial complexes and models, which are then shown equivalent

Later, chapter 4 offers a comprehensive introduction to the results presented in

Alcántara et al. (2019), a fundamental step for the usage of the topological approach on distributed computing for theoretical robotics. The models and the problems are presented, and the results discussed. The initial version of robot tasks in the literature is present at this part, however one will notice that multiple elements of its interpretation and definition are modified by the end of this work.

Following this base, the existing work and motivations for the study of exploration tasks and sensory limitations is exposed, also accompanied by the recent results and a proposed robot model. At last, we see an exercise on the topological representation of a simple robot task, with the full evolution of a single step in the case of two robots working in a three vertices line graph. This served as one base of intuition for the later formalization of tasks.

It is noticeable how these equivalences each bring different sets of tools for tackling with robotic tasks. Topological methods favor the derivation of impossibility results, as topological invariants are easier to verify, together with their consequences in the solvability framework. On the other hand, the logical approach is much more suited for verification, since they facilitate the reasoning of possibility results. Together, they vastly improve the tools available when analyzing missions of robots as tasks.

In the last chapter 5, a new understanding of robot tasks is exposed, together with the intuitions for it. This is followed by the actual formalization, which gives insights to a new approach still in development for the proof of guaranteed missions in swarm robotics. The key features, beyond a new solvability criteria, is its natural support for multiple types of robotic failures, such as crashes and freezes, for unreliable movement and limited sensing capabilities of the agents. This also extends the previous definition by supporting diverging tasks, such as exploration, since it allows for the representation of more specific and descriptive configurations.

A presentation of a part of this work has been done in a seminar of works related to the AID CIEDS FARO Project, at ENSTA Paris. One should note that the activities developed have undertaken a more formative path, as a preparation for a PhD project under the same title with École Polytechnique, focusing on broader foundations than in a different case.

6.2 Future work

The proposed formalization of robot tasks still has to be tested in a range of applications, even though problems of gathering and exploration were used as motivations during the construction of this new framework. The next envisioned step is to verify existing results in the literature using the new framework, ideally achieving the same conclusions. This should be done first with the most simple cases, and later evolved are more of the features described above are used, mainly with faulty robots and limited sensing.

The study of the modal logic equivalencies has just begun, and another future step is to explore other results to take advantage between the two domains. Specifically, the usage of multimodalities, i.e. both epistemic and temporal logic at the same time, is of great interest for the expression of robot tasks. Beyond the existing theory to be exploited, it is interesting the possibility of using the logic language for the description of missions, which can be much more easily translated to natural language.

The structures used for the proposed formalization fits the requirements for forming an algebra operating via endofunctors, and another idea is to explore this in order to define a “step” operator, a more formal way to describe robots evolving over time. This goes alongside the interest in multimodalities.

On the side of efforts approaching theoretical and practical results, one interest is to find a better graph discretization of continuous environments. Since all algorithms are reasoned based on a family of graphs, an immediate mapping would facilitate the interpretation and usage of guaranteed results. This choice will hardly be unique, but it would be an interesting advancement to define rules for it, according to the sensors available and the type of environment. This has been briefly explored with the usage of grid graphs for underwater maps.

At last, still aiming in a facilitated transition between theoretical and applied methods, the development of tools for visualization and verification would certainly help. In a first moment, the visualization of the subdivision of simplicial complexes would already help with the interpretation, given that those structures can grow quickly in size and complexity. Further work in the verification of tasks are also envisioned, with such a tool making possible a faster verification of the viability of missions at a very early stage of development. This is in part favored with the logical equivalences that are already suited for proving possibility results.

7 APPENDIX

7.1 Equivalence of categories

The idea of equivalent categories can be worked from the perspective of the 2-category Cat, the category of all categories, which has categories as objects, functors between them as morphisms and natural transformations as 2-morphisms. In this environment, the corresponding notion to isomorphism of objects will carry the idea of equivalence between categories, and this is the case of pairs of functors between categories that are inverse to each other up to natural isomorphism. In the analogy of "forgetfullness" of functors, those that belong to an equivalence relation forget neither properties nor structure between categories.

This means that we can specify two functors

$$F : \mathcal{C} \to \mathcal{D}$$

and

$$G : \mathcal{D} \to \mathcal{C}$$

that allow going from one category to another, presenting natural isomorphism (their composition relates to the identity functors):

$$F \circ G \simeq id_{\mathcal{D}} \quad G \circ F \simeq id_{\mathcal{C}}$$

An alternative to attest this equivalence is if the functors F and G are essentially surjective and fully faithful, meaning that they are surjective up to isomorphism and functions between its hom sets (the set of all morphisms) are bijective.

As an essentially surjective functor $F : \mathcal{C} \to \mathcal{D}$, for every object y of $\mathcal{D}$, there will be an object x of $\mathcal{C}$ that satisfies the isomorphism $F(x) \cong y$ in $\mathcal{D}$. The same has to be true for the functor $G : \mathcal{D} \to \mathcal{C}$.

As a fully faithul functor, $F : \mathcal{C} \to \mathcal{D}$ will have the following function to be bijective

$$F : \mathcal{C}(x, y) \to \mathcal{D}(F(x), F(y))$$

The key intuition is to have two categories presenting the same skeleton (without the fat, stripping down the representation of the objects), and that are the same as long as we look only at the morphisms.

With the equivalences established in the next sections, we will be able to think more concretely on the similar concepts that can be shared between domains.

7.1.1 Equivalence between pure simplicial complexes and Kripke frames.

An interesting development is the connection of epistemic logic and distributed computability, which is done via an equivalence between the *simplicial complexes* model, combinatorial topological structures used to represent distributed tasks, presented in section 2.1.2, and the *Kripke* model, presented in section 2.3.5.

Those constructions give raise to the following categories.

Category 7.1 *$\mathcal{S}_{\mathcal{A}}$ is a category of pure chromatic simplicial complexes on A with*

- *objects: pure chromatic simplicial complexes, where all maximal complexes are of the same dimension n and the coloring map is defined over A.*
- *morphisms: chromatic simplicial maps.*

Category 7.2 *$\mathcal{K}_{\mathcal{A}}$ is a category of proper Kripke frames over A with*

- *objects: proper Kripke frames, e.g. $M = \langle S, \sim\rangle$, $N = \langle T, \sim'\rangle$, where all states are distinguishable by at least one agent.*
- *morphisms: a morphism $f : M \to N$ means that there is mapping between the states S to T such that for all $a \in A$, for all $u, v \in S$, $u \sim_a v$ implies $f(u) \sim'_a f(v)$.*

From Goubault, Ledent and Rajsbaum (2018) we have the following theorem.

Theorem 7.1 *$\mathcal{S}_{\mathcal{A}}$ and $\mathcal{K}_{\mathcal{A}}$ are equivalent categories.*

This is proven via the constructions of two functors $F : \mathcal{S}_{\mathcal{A}} \to \mathcal{K}_{\mathcal{A}}$ and $G : \mathcal{K}_{\mathcal{A}} \to \mathcal{S}_{\mathcal{A}}$ as follows.

The first functor F takes a simplicial complex C and generates a Kripke frame $F(C) = \langle S, \sim\rangle$, where a set of states S is generated for each facet of C, and the equivalence relation $\sim_a$ for all $a \in A$ is obtained from the intersection of facets, i.e. for the facets X and Y of C, there is a relation $X \sim_a Y$ if $a \in \mathcal{X}(X \cap Y)$.

The chromatic simplicial mapping $f : C \to D$ in $\mathcal{S}_{\mathcal{A}}$ is respected via $F(f) : F(C) \to F(D)$, as for any $a \in A$ that is the coloring of a vertex in the intersection of facets in C will also be a coloring in the intersection of those facets after f, a consequence of them being rigid and never merging vertices, i.e. $\mathcal{X}(v) = a = \mathcal{X}(f(v))$, where $v \in X \cap Y$ and $f(v) \in f(X) \cap f(Y)$. Consequently, $f(X) \sim_a f(Y)$ holds and $F(f)$ will be a morphism between Kripke frames.

The second functor G takes a Kripke frame $M = \langle S, \sim \rangle$ defined over a set of agents $A = \{a_0, ..., a_n\}$ and generates a simplicial complex $G(M) = C$. The vertices of C will be equivalence classes, considering a *n*-simplex for each state $s \in S$, i.e. $V = \{v_i^s | s \in S, 0 \leq i \leq n\}$, and two vertices v_i^s and $v_i^{s'}$ that belong to the same equivalence class $[v_i^s] \in V/\sim_{a_i}$ if they correspond to equivalent states via the agent a_i, i.e. $s \sim_{a_i} s'$. The simplexes of C will be defined as $\{[v_0^s], ..., [v_n^s]\}$. The coloring $\mathcal{X}([v_k^s]) = a_i$ is well-defined as the agents in the same equivalence class have the same color. The facets of C will be all of the simplexes, as all states can be distinguished in the proper Kripke frame.

A morphism between Kripke frames $f : M \rightarrow N$ will be mapped to a chromatic simplicial map $G(f) : G(M) \rightarrow G(N)$ consistently as going from a vertex $[v_i^s] \in G(M)$ to $[v_i^{f(s)}] \in G(N)$ does not depend on the representative of the equivalence class to preserve the indistinguishability relation.

The composition of F and G defines isomorphisms, both in $\mathcal{K}_{\mathcal{A}}$ and $\mathcal{S}_{\mathcal{A}}$. This can be verified from the properties above being bijective relations, constituting a pair of adjoint functors.

Example 7.1 *Figure 7.1 shows an example of this equivalence. Each of the three states is transformed into a 2-dimensional simplex, facets composed of 3 vertices, one for each agent. It is noticeable that the two states that are indistinguishable only by agent b, in black, are the triangles connected via the black vertex. The other two states, indistinguishable by agents g and w, are connected via the grey and white verticces.*

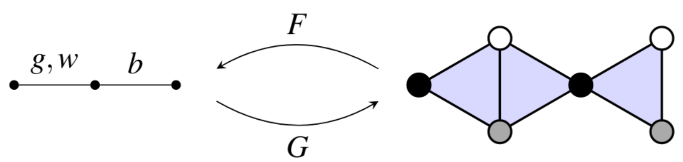


Figure 7.1 – Equivalence via functors of a proper Kripke frame (left) and a pure chromatic simplicial complex (right). From Goubault, Ledent and Rajsbaum (2018).

7.1.2 Equivalence between simplicial models and Kripke models

Extending the concepts of simplicial complexes and Kripke frames to also contain information about values held by the agents, we arrive at the simplicial models and Kripke models. The information are *atomic propositions*, defined as $AP = \{p_{a,x} | a \in A, x \in \mathcal{V}\}$, and $\mathcal{V}$ is a set of countable values. The interpretation for $p_{a,x}$ is true if agent a hold the value x.

Definition 7.1 *A* simplicial model *is a triple $M = \langle C, \mathcal{X}, l \rangle$, where C and $\mathcal{X}$ define a pure chromatic simplicial complex, and $l : \mathcal{V}(C) \to \mathcal{P}(AP)$ a labeling function that associates a set of atomic propositions to each vertex of the simplicial complex. The atomic propositions are related to the agent represented in that vertex, i.e. $\mathcal{X}(v)$, where $l(v) \subseteq AP_{\mathcal{X}(v)}$.*

Definition 7.2 *A* Kripke model *is a triple $M = \langle S, \sim, L \rangle$, where S and $\sim$ define a proper Kripke frame, and $L : S \to \mathcal{P}(AP)$ retrieves the set of atomic propositions that are true in a state $s \in S$. The kripke model will be* local *if for every agent $a \in A$, all equivalent states for a contain the same set of atomic propositions concerning a, i.e. $s \sim_a s'$ implies $L(s) \cap AP_a = L(s') \cap AP_a$.*

From those models, we can define the following categories.

Category 7.3 *$\mathcal{SM}_{A,AP}$ is the category of simplicial models over the set of agents A and atomic propositions AP with:*

- *objects: simplicial models.*
- *morphisms: chromatic simplicial mapping that preserver labeling, i.e. given $f : M \to M'$, $l'(f(v)) = l(v)$.*

Category 7.4 *$\mathcal{KM}_{A,AP}$ is the category of local proper Kripke models with:*

- *objects: local proper Kripke models.*
- *morphisms: morphisms between Kripke frames that preserve labeling, i.e. given $f : M \to M'$, for every state $s \in S$, $L'(f(s)) = L(s)$.*

Which are stated equivalent in the second theorem from Goubault, Ledent and Rajsbaum (2018).

Theorem 7.2 *$\mathcal{SM}_{A,AP}$ and $\mathcal{KM}_{A,AP}$ are equivalent categories.*

The functors $F : \mathcal{SM} \to \mathcal{KM}$ and $G : \mathcal{KM} \to \mathcal{SM}$ are used to navigate between those two categories, and they use the same operations explained in section 7.1.1 for the underlying simplicial complexes and Kripke frames.

With F, given a simplicial model $M = \langle C, \mathcal{X}, l\rangle$, we associate a Kripke model $F(M) = \langle \mathcal{F}(C), \sim, L\rangle$. As we have to associate the labeling of the simplicial complex to sets of true atomic propositions of a state, where the labeling $L(X)$ of a facet $X \in \mathcal{F}(C)$ is given by the union of the labeling of all vertices belonging to that facet, i.e. $L(X) = \bigcup_{v \in X} l(v)$. Note that facets are translated into states in the Kripke model. It will be local as equivalent states for an agent a, $X \sim_a Y$, come from facets that share a vertex, and the information in the labeling of that vertex will be present in both states, i.e. $L(X) \cap AP_a = L(Y) \cap AP_a = l(v)$.

With G, given a Kripke model $M = \langle S, \sim, L\rangle$, we associate a simplicial model $G(M) = \langle \mathcal{G}(S), \mathcal{X}, l\rangle$. Here, we have to define a labeling function according to the true atomic propositions in each state $s \in S$, considering that they are mapped to *n*-simplexes $\{v_0^s, ..., v_n^s\}$, according to the number of agents n. The label of each vertex v_i^s, which corresponds to the view of the state s by agent a_i, will be the true propositions related to a_i in s, i.e. $l(v_i^s) = L(s) \cap AP_{a_i}$. Since the Kripke model is local, two equivalent states according to an agent will always have the same labeling.

Again, F and G constitute two adjoint functors with the properties shown above and the composition of them lead to isomorphisms.

Example 7.2 *We can see in figure 7.2 another example of this equivalence. Here, each of the two agents has a binary value of 0 or 1 and knows its own state, but doesn't know the one from the other agent. This means that for each possible value of an agent, there are two equally possible states for the other agent.*

In the Kripke model, this is represented with the shared states (pairs of binary values) that are indistinguishable according to the agents in the connecting edges. For example, agent w cannot distinguish between the global statews 01 *and* 11*, as it knows to have the value* 1*, while g may have* 0 *or* 1*. This is translated in the simplicial model with an edge, a 1-dimensional complex, for each global state, and the colored vertices at the extremes representing the agents. Their labels are their own state, so an edge with the global state* 01 *will connect a gray vertex labeled* 0 *and a white vertex labeled* 1*.*

As theorem 7.2 states the equivalence between simplicial and Kripke models, it is possible to express the semantics present in Kripke models in terms of simplicial models.

Definition 7.3 *The truth value of some ϕ belonging to the epistemic logic using agents A and atomic propositions AP, can be evaluated in a given epistemic state (M, X), where*

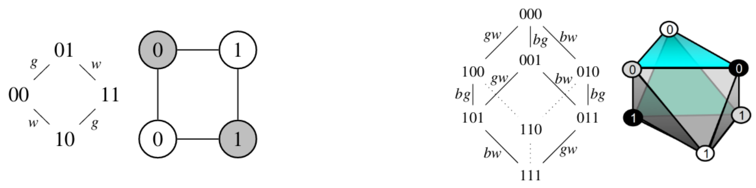


Figure 7.2 – Equivalence via functors of a local proper Kripke model (left of each pair) and a simplicial model (right of each pair). From Goubault, Ledent and Rajsbaum (2018).

$M = \langle C, \mathcal{X}, l \rangle$ is a simplicial model and X is a facet of that simplicial model. This truth value is defined as follows:

$M, X \models p$	iff	$p \in l(X)$
$M, X \models \neg\phi$	iff	$M, X \not\models \phi$
$M, X \models \phi \wedge \psi$	iff	$M, X \models \phi$ *and* $M, X \models \psi$
$M, X \models K_a\phi$	iff	*for all* $Y \in F(C), a \in \mathcal{X}(X \cap Y)$ *implies* $M, Y \models \phi$

And this leads to the following proposition.

Proposition 7.1 *Given a simplicial model M and a facet X, $M, X \models \phi$ iff $F(M), X \models_{\mathcal{K}} \phi$. Conversely, given a local proper Kripke model N and state s, $N, s \models_{\mathcal{K}} \phi$ iff $G(N), G(s) \models \phi$, where $G(s)$ is the facet $\{v_0^s, ..., v_n^s\}$ of $G(N)$.*

REFERENCES

ALCáNTARA, M. et al. The topology of look-compute-move robot wait-free algorithms with hard termination. **Distributed Computing**, v. 32, n. 3, p. 235–255, jun. 2019. ISSN 1432-0452. Available from Internet: <https://doi.org/10.1007/s00446-018-0345-3>.

ARMENTA-SEGURA, J.; RAJSBAUM, S.; LEDENT, J. Two-agent Approximate Agreement from an Epistemic Logic perspective. p. 16, 2020.

BARR, M.; WELLS, C. **Category Theory for Computing Science**. [s.n.], 1990. ISBN 978-0-13-120486-7. Available from Internet: <https://dl.acm.org/doi/10.5555/92134>.

BENTHEM, J. v. **Modal logic for open minds**. Stanford, California: Center for the Study of Language and Information, 2010. (CSLI lecture notes, no. 199). OCLC: ocn457148948. ISBN 978-1-57586-599-7 978-1-57586-698-7 978-1-57586-598-0.

BONNET, F.; DEFAGO, X. Exploration and Surveillance in Multi-robots Networks. In: **2011 Second International Conference on Networking and Computing**. [S.l.: s.n.], 2011. p. 342–344.

BOROWSKY, E. et al. The BG distributed simulation algorithm. **Distributed Computing**, v. 14, n. 3, p. 127–146, jul. 2001. ISSN 1432-0452. Available from Internet: <https://doi.org/10.1007/PL00008933>.

BRAMAS, Q.; LAFOURCADE, P.; DEVISMES, S. Optimal Exclusive Perpetual Grid Exploration by Luminous Myopic Opaque Robots with Common Chirality. In: **International Conference on Distributed Computing and Networking 2021**. Nara Japan: ACM, 2021. p. 76–85. ISBN 978-1-4503-8933-4. Available from Internet: <https://dl.acm.org/doi/10.1145/3427796.3427834>.

CARNIELLI, W.; PIZZI, C. **Modalities and Multimodalities**. [S.l.: s.n.], 2008. OCLC: 1144208333. ISBN 978-1-4020-8590-1.

DAS, S. et al. Autonomous mobile robots with lights. **Theoretical Computer Science**, v. 609, p. 171–184, jan. 2016. ISSN 03043975. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S0304397515008476>.

DATTA, A. K. et al. Ring Exploration by Oblivious Agents with Local Vision. In: **2013 IEEE 33rd International Conference on Distributed Computing Systems**. [S.l.: s.n.], 2013. p. 347–356. ISSN: 1063-6927.

DATTA, A. K. et al. Ring Exploration by Oblivious Robots with Vision Limited to 2 or 3. In: HIGASHINO, T. et al. (Ed.). **Stabilization, Safety, and Security of Distributed Systems**. Cham: Springer International Publishing, 2013. (Lecture Notes in Computer Science), p. 363–366. ISBN 978-3-319-03089-0.

DITMARSCH, H. van et al. Knowledge and simplicial complexes. **arXiv:2002.08863 [cs]**, feb. 2020. ArXiv: 2002.08863. Available from Internet: <http://arxiv.org/abs/2002.08863>.

FAGIN, R. (Ed.). **Reasoning about knowledge**. Cambridge, Mass: MIT Press, 1995. ISBN 978-0-262-06162-9.

FAJSTRUP, L. et al. **Directed Algebraic Topology and Concurrency**. Cham: Springer International Publishing, 2016. ISBN 978-3-319-15397-1 978-3-319-15398-8. Available from Internet: <http://link.springer.com/10.1007/978-3-319-15398-8>.

FLOCCHINI, P.; PRENCIPE, G.; SANTORO, N. (Ed.). **Distributed Computing by Mobile Entities: Current Research in Moving and Computing**. Cham: Springer International Publishing, 2019. (Lecture Notes in Computer Science, v. 11340). ISBN 978-3-030-11071-0 978-3-030-11072-7. Available from Internet: <http://link.springer.com/10.1007/978-3-030-11072-7>.

FLOCCHINI, P. et al. Gathering of asynchronous robots with limited visibility. **Theoretical Computer Science**, v. 337, n. 1-3, p. 147–168, jun. 2005. ISSN 03043975. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S0304397505000149>.

GOTOH, T. et al. Exploration of dynamic networks: Tight bounds on the number of agents. **Journal of Computer and System Sciences**, v. 122, p. 1–18, dec. 2021. ISSN 0022-0000. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0022000021000416>.

GOUBAULT, E. **Private notes on algebra, coalgebras and temporal-epistemic logics**. 2021.

GOUBAULT, ; LEDENT, J.; RAJSBAUM, S. A Simplicial Complex Model for Dynamic Epistemic Logic to study Distributed Task Computability. **Electronic Proceedings in Theoretical Computer Science**, v. 277, p. 73–87, sep. 2018. ISSN 2075-2180. ArXiv: 1809.03095. Available from Internet: <http://arxiv.org/abs/1809.03095>.

GOUBAULT, ; LEDENT, J.; RAJSBAUM, S. A Simplicial Model for KB4n: Epistemic Logic with Agents that May Die. p. 20, 2022.

GUERRAOUI, R.; MAURER, A. Gathering with extremely restricted visibility. **arXiv:1906.06239 [cs]**, jun. 2019. ArXiv: 1906.06239. Available from Internet: <http://arxiv.org/abs/1906.06239>.

HERLIHY, M.; KOZLOV, D. N.; RAJSBAUM, S. **Distributed computing through combinatorial topology**. Amsterdam: Morgan Kaufmann, is an imprint of Elsevier, 2014. ISBN 978-0-12-404578-1.

HERLIHY, M.; SHAVIT, N. The topological structure of asynchronous computability. **Journal of the ACM**, v. 46, n. 6, p. 858–923, nov. 1999. ISSN 0004-5411, 1557-735X. Available from Internet: <https://dl.acm.org/doi/10.1145/331524.331529>.

KIRKPATRICK, D. et al. Separating Bounded and Unbounded Asynchrony for Autonomous Robots: Point Convergence with Limited Visibility. In: **Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing**. Virtual Event Italy: ACM, 2021. p. 9–19. ISBN 978-1-4503-8548-0. Available from Internet: <https://dl.acm.org/doi/10.1145/3465084.3467910>.

KNIGHT, S. The Epistemic View of Concurrency Theory. p. 217, sep. 2013.

LANE, S. M. **Categories for the working mathematician**. New York: Springer-Verlag, 1971. (Graduate texts in mathematics, 5). ISBN 978-0-387-90035-3 978-0-387-90036-0 978-3-540-90036-8.

LEDENT, J. Brief Announcement: Variants of Approximate Agreement on Graphs and Simplicial Complexes. In: **Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing**. Virtual Event Italy: ACM, 2021. p. 427–430. ISBN 978-1-4503-8548-0. Available from Internet: <https://dl.acm.org/doi/10.1145/3465084.3467946>.

LUNA, G. A. D. et al. TuringMobile: a turing machine of oblivious mobile robots with limited visibility and its applications. **Distributed Computing**, v. 35, n. 2, p. 105–122, abr. 2022. ISSN 0178-2770, 1432-0452. Available from Internet: <https://link.springer.com/10.1007/s00446-021-00406-6>.

NAGAHAMA, S.; OOSHITA, F.; INOUE, M. Ring Exploration of Myopic Luminous Robots with Visibility More Than One. In: GHAFFARI, M. et al. (Ed.). **Stabilization, Safety, and Security of Distributed Systems**. Cham: Springer International Publishing, 2019. (Lecture Notes in Computer Science), p. 256–271. ISBN 978-3-030-34992-9.

NAGAHAMA, S.; OOSHITA, F.; INOUE, M. Terminating Grid Exploration with Myopic Luminous Robots. In: **2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)**. [S.l.: s.n.], 2021. p. 586–595.

NAKAI, R.; SUDO, Y.; WADA, K. Asynchronous Gathering Algorithms for Autonomous Mobile Robots with Lights. In: JOHNEN, C.; SCHILLER, E. M.; SCHMID, S. (Ed.). **Stabilization, Safety, and Security of Distributed Systems**. Cham: Springer International Publishing, 2021. (Lecture Notes in Computer Science), p. 410–424. ISBN 978-3-030-91081-5.

OOSHITA, F.; TIXEUIL, S. Ring exploration with myopic luminous robots. **Information and Computation**, p. 104702, jan. 2021. ISSN 08905401. Available from Internet: <https://linkinghub.elsevier.com/retrieve/pii/S0890540121000171>.

RIEHL, E. **Category Theory in Context**. [S.l.: s.n.], 2016.

SMITH, P. **A Gentle Introduction to Category Theory**. [S.l.: s.n.], 2018.