Opinions, Lies and Knowledge. An Algebraic Approach to Mobility of Information and Processes

Yamil Salim Perchy

NNT : 2016SACLX059

Thèse de Doctorat

de

l'Université Paris-Saclay

préparée à

l'École Polytechnique

École doctorale n°580

Sciences et technologies de l'information et de la communication

Spécialité de doctorat : Informatique

Par

M. Salim Perchy

Opinions, Lies and Knowledge. An Algebraic Approach to Mobility of Information and Processes

Thèse présentée et soutenue à Palaiseau, le 4 Octobre 2016 :

Composition du Jury :

Mme. Putot, Sylvie	Professeure, École Polytechnique	Présidente du Jury
M. Falaschi, Moreno	Professeur, Università di Siena	Rapporteur
M. Mousavi, M. Reza	Professeur, Högskolan i Halmstad	Rapporteur
M. Rocha, Camilo	Professeur, Universidad Javeriana	Examinateur
M. Valencia, Frank	Chercheur, CNRS - LIX	Directeur de thèse
M. Haar, Stefan	Chercheur, INRIA - LSV,	Co-directeur de thèse

Résumé

La notion de *système de contraintes* (cs – selon l'acronyme anglais) est un concept central aux formalismes de *la théorie de la concurrence* tels que les algèbres de processus pour la programmation concurrente par contraintes. Les systèmes de contraintes sont souvent représentés par des *treillis* : ses éléments, appelées contraintes, représentent des informations partiales tandis que l'ordre du treillis correspond à des implications. Récemment, une notion appelée "système de contraintes spatiales à n-agents" a été développée pour représenter l'information dans la programmation concurrente par contraintes où les systèmes sont multi-agents et spatialement distribués.

D'un point de vue informatique, un système de contraintes spatiales peut être utilisé pour spécifier l'information partiale contenue dans l'espace d'un certain agent (*information locale*). D'un point de vue épistémique, un cs spatial peut être utilisé pour représenter l'information qui est considérée vrai pour un certain agent (*croyance*). Les systèmes de contraintes spatiales, néanmoins, ne fournissent pas de mécanismes pour la spécification de la mobilité de l'information ou des processus d'un espace à un autre. La mobilité de l'information est un aspect fondamental des systèmes concurrents.

Dans cette thèse nous avons développé la théorie des systèmes de contraintes spatiales avec des opérateurs pour spécifier le déplacement des informations et processus entre les espaces. Nous étudions les propriétés de cette nouvelle famille de systèmes de contraintes et nous illustrons ses applications. Du point de vue calculatoire, ces nouveaux opérateurs nous apportent de l'*extrusion* d'informations et/ou des processus, qui est un concept central dans les formalismes pour la communication mobile. Du point de vue épistémique, l'extrusion correspond à une notion que nous avons appelé *énon-*

ciation ; une information qu'un agent souhaite communiquer à d'autres mais qui peut être inconsistante avec les croyances de l'agent même. Des énonciations peuvent donc être utilisées pour exprimer des notions épistémiques tels que les canulars ou les mensonges qui sont fréquemment utilisés dans les réseaux sociaux.

Globalement, les systèmes de contraintes peuvent exprimer des notions épistémiques comme la *croyance*/*énonciation* et la *connaissance* en utilisant respectivement une paire de fonctions espace/extrusion qui représentent l'*information locale*, et un opérateur spatial dérivé qui représente l'*information globale*. Par ailleurs, nous montrons qu'en utilisant une construction précis de systèmes de contraintes, nous pouvons aussi représenter la notion du temps comme une imbrication des espaces qui représentent eux-même une séquence d'instances.

Abstract

The notion of *constraint system (cs)* is central to declarative formalisms from *concurrency theory* such as process calculi for concurrent constraint programming (ccp). Constraint systems are often represented as *lattices*: their elements, called *constraints*, represent partial information and their order corresponds to entailment. Recently a notion of *n-agent spatial cs* was introduced to represent information in concurrent constraint programs for spatially distributed multi-agent systems. From a *computational* point of view a spatial constraint system can be used to specify partial information holding in a given agent's space (*local information*). From an *epistemic* point of view a spatial cs can be used to specify information that a given agent considers true (*beliefs*). Spatial constraint systems, however, do not provide a mechanism for specifying the mobility of information/processes from one space to another. Information mobility is a fundamental aspect of concurrent systems.

In this thesis we develop the theory of spatial constraint systems with operators to specify information and processes moving between spaces. We investigate the properties of this new family of constraint systems and illustrate their applications. From a computational point of view the new operators provide for process/information *extrusion*, a central concept in formalisms for *mobile communication*. From an epistemic point of view extrusion corresponds to what we shall call *utterance*; a piece of information that an agent communicates to others but that may be inconsistent with the agent's beliefs. Utterances can be used to express instances of epistemic notions such as *hoaxes* or intentional *lies* which are common place in social media.

On the whole, constraint systems can express the epistemic notions of *belief*/*utterance* and *knowledge* by means of respectively, a space/extrusion function pair that specifies *local information* and a derived spatial operator that specifies *global information*. We also show that, by using a specific construction of our constraint systems, we can encode the notion of *time* as an arbitrary nesting of spaces representing a *sequence of instances*.

Acknowledgments

There is always a bittersweet taste when one writes acknowledgments. Just after starting a PhD, one sees its end in a very distant, fuzzy and dystopian way, when ending a PhD, one sees its beginning in a vivid, nostalgic and, as contrary as it may appear, utopian way. This is a testament that all these years dedicated to science can radically change your attitude, your perspective and your behaviors. Not only you learn how to write papers and research proposals, it is a social challenge that forces you to know yourself better, a mental test that teaches you how to get fortitude out of nothing.

And even if the world outside your lab minimizes this journey to either a list of publications or a two-page *résumé de thèse* when applying to a job out there in the jungle, none but you know its true cost. A bill that is passed with a total not in terms of time and money, but in terms (whether debited or credited) described in friendships, happiness, anxiety, obscurity, experiences, landscapes, family members, pictures, disappointments, expectations, and so on. All this whistle one is in this period of almost total quiescence in terms of finding a proper north. Nonetheless, as journeys are usually done with other travelers (at least during the time when their paths run in the same direction), this one was no exception and it is one's responsibility to mention these fellow adventurers.

I would like to express here my gratitude to my mother Janeth, my father Yamil and my sister Faryde. These travelers, though far away, are still going in the same direction providing me with wisdom, love, and above all, unconditional support. I also wish to praise my soulmate Laura, a big part of my life and dreams and a big supporter of all my decisions. No less decisive, I want to acknowledge the importance of my supervisor Frank, the one responsible for this big opportunity and for passing onto me the secrets of science. I'm also indebted with Stefan and

Camilo for their supervision and big scientific advices. In these three years I made part of Comète, I couldn't ask for a better team and because of this I would like to thank Catuscia for her immense appreciation towards us, she makes every member of the team feel like he matters. Bibek, Michell, Luis, Yusuke, Marco, Ehab, Joris, Benjamin, Nico, Lili, and Kostas all were great teammates and colleagues. At the lab, I made the great acquaintance of Tomer Libal and his family, I wanna thank him for his friendship, chats and pizza evenings. I also thank professor Jean-Marc Steyaert with whom I chatted on the halls when staying late to work.

While I was living in France, I had the huge luck of meeting Guillaume Tirefort, one of those friends you don't cross paths with every day, I want to show my utmost appreciation for his friendship that I'm sure will last for many years more. I also wanna show my gratitude to the RPG group with its fun sessions and jokes; David, Alberto, Ben, Chiara, Enrico, Adam, Philippe, Nicolas, Spyros and Bethany. It is also vital that I acknowledge the friendship offered to me by Fereshteh Asgari, Vanessa Molina, Kaja Osiecka, Kamila Piotrowska, Fatima Jardali, Yu Chao and Mme. Traore; they are remarkable women I had the luck of meeting here and I will remember them for years to come. My longtime friend Francisco, who even with the time difference, had the time and disposition to always be there virtually and geek out. Though I didn't see them in the flesh, whenever I needed a laugh I could always count on Mat & Pat and JonTron.

All experiences in a journey do not necessarily come from people and, because I'm a great admirer of the 7th and 8th arts, I want to compile here a list of films and video-games that, during this three-year period, made an impact and/or gave comfort when solitude was the only option.

- One Eyed Girl (2014)
- The Rover (2014)
- Zulu (2013)
- Cold Fear (2005)
- The Fall (2006)
- Ender's Game (2013)
- The Elephant Man (1980)
- The Tree of Life (2011)
- Portal 2 (2011)
- Beneath a Steel Sky (1994)
- Katmandú, un espejo en el cielo (2011)
- The Grey (2011)
- All is Lost (2013)
- Alpha Protocol (2010)
- Blackbird (2012)
- The Thin Red Line (1998)
- Missing: An Interactive Thriller (2015)
- The Unknown (1927)

— Deep Fear (1998)
— Eternal Sunshine of the Spotless Mind (2004)
— Me and Earl and the Dying Girl (2015)
— Shutter Island (2011)
— El Secreto de sus Ojos (2009)
— Respire (2014)
— Un Illustre Inconnu (2014)
— V/H/S/2 (2013)
— Sea Fog (2014)
— Tel Père, Tel Fils (2013)
— Room (2015)
— Relatos Salvajes (2015)
— Fin (2012)
— La Isla Mínima (2014)
— Echo (2014)
— Casshern (2004)
— Minuscule - La Vallée des fourmis perdues (2013)
— The Tribe (2014)
— Whiplash (2014)
— Los Últimos Días (2013)
— Everest (2015)
— Albator, Corsaire de l'Espace (2013)
— Solace (2015)
— Babysitting (2014)
— Wild (2014)
— The Hunt (2012)
— I Am Alive (2012)
— Pawn Sacrifice (2014)
— Stockholm (2013)
— The Forest (2016)
— My Sassy Girl (2008)
— Demolition (2016)
— J.U.L.I.A. Among the Stars (2014)
— The Boys are Back (2009)
— Unfriended (2014)
— Stranger Things (2016)
— Jack et la Mécanique du Cœur (2014)
— Remember Me (2013)
— Reign Over Me (2007)
— Marathon (1994)
— Equals (2015)

Last but not remotely least, I would like to thank INRIA and the Government of France for completely funding my PhD. I will be forever indebted to the people of France.

Contents

Chapter 1

Introduction

A sound, conscious, and thorough study about digital information is always a welcomed effort. In today's digital age, this intrinsic motivation is not freely granted by the apparent pervasiveness of the information revolution and its ever-increasing scale. It is intrinsic because of the actual democratization of information, and freely granted due to the (virtually) zero cost now associated with its appropriation. Thus, it seems rather disparate that a work of scientific research attributes its motivation to a political/economical impetus based on people's epistemic relation with information and participation thereof. This I think, merits a philosophical introduction to an utterly scientific text.

The grasping of philosophical meanings is, more often than not, done by unearthing and analyzing ancient erudites. But with a phenomenon as modern as the information revolution, it is only fair to quote thinkers from our generation. And for this, I turn to a relatively niche film from 1995 called *Ghost in the Shell*, where our heroin *Major Kusanagi* is in a cat-and-mouse chase after an elusive hacker called *The Puppet Master*, purportedly capable of hacking even into human bodies. When they finally meet face-to-face and he is asked for the reasoning behind his actions, *The Puppet Master* utters the following snippet amid a long monologue:

> ‘‘It can also be argued that DNA is nothing more than a program designed to preserve itself. Life has become more complex in the overwhelming sea of information. And life,

> when organized into species, relies upon genes to be its memory system. So, man is an individual only because of his intangible memory... and memory cannot be defined, but it defines mankind. The advent of computers, and the subsequent accumulation of incalculable data has given rise to a new system of memory and thought parallel to your own. Humanity has underestimated the consequences of computerization."

Although subjective, it can be argued that in the context of the information revolution the individuality of man directly depends on his knowledge. However, in the above discourse one thing remains objective and undisputed and that is the underestimation of the information revolution. Even so, *The Puppet Master* (or the script writer for that matter) wisely avoids mentioning what the consequences of this revolution are. We need not venture much into the future, only six years laters when in 2001 the video game *Metal Gear Solid 2: Sons of Liberty*, known for its long and philosophical radio conversations between the main characters, dared to answer part of that question. In this 8th art [Per15] opus we have our manly spy hero *Solid Snake*, trying to expose the existence of a secret state-of-the-art mobile bipedal nuclear tank. With the remote help of the geek, glass-wearing computer genius *Otacon*, they discuss at one point:

Otacon: When the photos are in, we'll put them online and blow this whole thing wide open on the Web.

Snake: Don't you think that the authorities will just shut us down?

Otacon: Probably. But it won't matter – there'll be mirror sites spawning within minutes after those images go up. We won't even have to ask; people will be grabbing the pictures. There's no way anyone can stop it. Information has a life of its own, and as long as it lasts, it'll keep existing – even if it has to change its form or location.

Snake: Life?

Otacon: Yeah – the desire of people to learn. The demand for data.

Snake: That's called idle curiosity in my book.

Otacon: Everyone's got some of that. And sometimes, it can pave the way for truth. That's what I believe.

Note that *Otacon*, apart from describing consequences of the information revolution, also ventures into moral judgements that are certainly outside the scope of any computer science text like this one. However, I (as author of this document) believe that this research thesis might provide men with mathematical and technological means for studying, and therefore gauging the consequences of this whole constant inertia of digital information (be it in its epistemic or doxastic direction). And what better motivation to do it than to stop people like *The Puppet Master* or to help scientists like *Otacon*.

Motivation

Epistemic, mobile and spatial behavior are common place in today's distributed systems. The intrinsic *epistemic* nature of these systems arises from social behavior. People are familiar with digital systems where users share their *beliefs*, *opinions* and even intentional *lies* (hoaxes). Also, systems modeling decision behavior must account for the partial dependance on the results of interacting with others within a social context. The courses of action stemming from some agent decision result not only from the rational analysis of a particular situation but also from the agent beliefs or information born from interactions with other participants involved in that situation. Appropriate performance in these social contexts requires the agent to form beliefs about the beliefs of others. Spatial and mobile behavior is exhibited by apps and data moving across (possibly nested) spaces defined by, for example, friend circles, groups, and shared folders. Thus we believe that a solid understanding of the notion of *space* and *spatial mobility* as well as the flow of epistemic information is relevant in any model of today's distributed systems.

Declarative formalisms of concurrency theory such as process calculi for *concurrent constraint programming* (ccp) [SRP91] were designed to give explicit access to the concept of partial information and, as such, have close ties with logic [PSSS93, MPSS95]. This makes them ideal for the incorporation of epis-

temic and spatial concepts by expanding the logical connections to include *multi-agent modal logic* [Kri63]. In fact, the sccp calculus [KPPV12] extends ccp with the ability to define local computational spaces where agents can store epistemic information and run processes.

Problem: Spatial and Epistemic Mobility

Context

Despite being able to express meaningful epistemic and spatial phenomena such as belief, local and global information, the sccp calculus does not provide a mechanism to intentionally *extrude* information or processes from local spaces. Such a mechanism would allow sccp to express the transfer of epistemic information from one space into another.

The notion of *constraint system (cs)* is central to ccp and other declarative formalisms such as (concurrent) constraint logic programming (clp). All ccp calculi (including sccp) are parametric in a cs that specifies partial information upon which programs (processes) may act. A cs is often represented as a *complete lattice* $(Con, \sqsubseteq)$. The elements of Con, the *constraints*, represent partial information and we shall think of them as being *assertions*. The order $\sqsubseteq$, the join $\sqcup$, the bottom *true* and the top *false* of the lattice correspond respectively to entailment, conjunction, the empty information and the join of all (possibly inconsistent) information.

Constraint systems provide the domains and operations upon which the semantic foundations of ccp calculi are built. As such, ccp operations and their logical counterparts typically have a corresponding elementary construct or operation on the elements of the constraint system. In particular, parallel composition and conjunction correspond to the *join* operation, and existential quantification and local variables correspond to a cylindrification operation on the set of constraints [SRP91].

Similarly, the notion of computational space and the epistemic notion of belief in sccp [KPPV12] correspond to a family of functions $[\cdot]_i : Con \to Con$ on the elements of the constraint system Con. These functions are called *space functions*. From a computational point of view the assertion (constraint) $[c]_i$ specifies that c

resides within the space of agent i. From an epistemic point of view, the assertion $[c]_i$ specifies that agent i considers c to be true (i.e. in the world of agent i assertion c is true). Both intuitions convey the idea of c being local (subjective) to agent i.

It is therefore natural to assume that a mechanism for extrusion in ccp ought to have a corresponding semantic concept in constraint systems. Furthermore, by incorporating extrusion directly in constraint systems, the concept may become available not only to sccp but also to other declarative constraint-based formalisms.

This Thesis: Algebraic Structures for Extrusion and Epistemic Reasoning

Our main goal is to investigate algebraic operations in constraint systems that help provide semantic foundations for extrusion and mobility of information. Systems where data moves across a given structure of information are now commonplace, and applications such as social networks, forums or any other that organizes its information in a defined hierarchy are among these systems. In practice, the nature of this information can be reviews, opinions, news, etc., which in turn belong to a certain entity, e.g. users, agents, applications. This relation of ownership can be conceptualized as *space*, and thus a clear understanding of information in spaces and its movement across them is pertinent to the study of these systems.

From a computational point of view, the new operations will allow us to specify mobile behavior as constraints. Consequently, this can help us model and analyze scenarios like process mobility and communicating agents and at the same time, detect potential offensive or intrusive behavior that may endanger the integrity or veracity of the information therein. Additionally, with a sound theory for mobility and multi-agent phenomena, we can enumerate important properties inherent to the systems and subsequently to all actors in them.

From a logical point of view, the operators will allow us to specify epistemic concepts such as opinions and intentional lies or even express properties regarding time. This way we can characterize well-stablished logical formalisms for epistemic behavior or time directives and reason about specifications made in them. As such, we are then able to describe the mobility of information as a concept inherent to a

specific context like belief, knowledge, utterance, future or past. We believe that an algebraic characterization of all these concepts is a novel approach that will help analyse or prevent behavior that might be undesired in a specification.

Our Contributions

In this work we generalize the underlying theory of spatial constraint systems by adding *extrusion* functions to their structure. We show that spatial constraint systems provide for the specification of spatial mobility and epistemic concepts such as belief, utterance and lies. We shall also show that the spatial theory of sccp [KPPV12], which captures belief, can also capture an epistemic notion of knowledge. This latter contribution is consistent with our goal of using algebraic spatial structures to capture epistemic behavior. Furthermore, we show that the concepts of future and past can be represented likewise by space and extrusion in order to express time properties.

Our main contributions can be summarized and structured as follows.

1. *E*xtrusion as the right inverse of space. We shall first introduce a family of functions $\uparrow_i$, called *extrusion functions*. Computationally, $\uparrow_i$ can be used to intentionally extrude information from within a space $[\cdot]_i$. Epistemically, $\uparrow_i$ can be used to express *utterances* by agent i. We shall put forward the notion of extrusion/utterance as the *right inverse* of space/belief. Under this interpretation we obtain

$$[c \sqcup \uparrow_i e]_i = [c]_i \sqcup e.$$

   This equation illustrates the extrusion of e from the space of agent i and it is reminiscent of *subjective mobility* in the ambient calculus [CG98]. By building upon concepts of Heyting Algebra, we will illustrate meaningful spatial and epistemic behaviors. In particular, *program mobility* and *intentional lies (hoaxes)*, i.e., utterance of statements by a given agent that are inconsistent with its beliefs.

2. *T*he Extrusion Problem. We consider the following problem: Given a space function $[\cdot]_i$ derive an extrusion (function) $\uparrow_i$ for it. We will provide canonical constructions of extrusion for surjective space functions that satisfy limit

conditions such as Scott-continuity. We shall also prove an impossibility result for the existence of join-preserving extrusion for surjective and Scott-continuous space functions.

3. *P*roperties of Extrusion. We will also investigate distinctive properties of space and extrusion functions. We will show that space functions that admit extrusion are necessarily space consistent: $[\mathit{false}]_i = \mathit{false}$. This corresponds to the Consistency Axiom of Epistemic (Doxastic) logic stating that no agent believes the false statement. We shall show that extrusion functions are *order embeddings*, and that injective spaces are *order automorphisms* (hence they preserve all limits). We shall also identify necessary and sufficient conditions under which space and extrusion form a *Galois connection*: namely a correspondence of the form $[c]_i \sqsubseteq d \Leftrightarrow c \sqsubseteq \uparrow_i d$.

4. *A* logic of Belief and Utterance. As an application of the above-mentioned contributions we show how to derive extrusion for a previously-defined instance of spatial constraint systems, namely, Kripke spatial cs [KPPV12]. We also derive the semantics for a logic of belief with reverse modalities by interpreting its formulae as elements in the Kripke spatial cs with extrusion. We can then show how to express instances of epistemic notions such as utterances and lies directly in the syntax of this logic. We conclude by showing that belief and utterance in this logic also form a Galois connection. Roughly speaking, this connection allows us to reduce the implication *of* belief from/to implication *by* utterance.

5. *K*nowledge in Terms of Global Space. We shall represent knowledge by using a derived spatial operation that expresses global information. The new representation is shown to obey the epistemic principles of the logic for knowledge S4. We also show a sound and complete spatial cs interpretation of S4 formulae. In previous work [KPPV12] spatial constraint systems were required to satisfy additional properties in order to capture S4 knowledge. Namely, space functions had to be closure operators. Here we will show that S4 knowledge can be captured in spatial constraint systems without any further requirements.

6. *E*xpressing time properties. To apply our theory to different scenarios than

those of epistemic interpretations, we take Linear Time Temporal Logic (LTL) [PM92] and characterize it using a special type of constraint system we call Time Constraint System. Here, we regard the future operator of LTL as the space function of the cs and proceed to derive two versions of the extrusion function. We show that one corresponds to the past operator of LTL and the other to a weak version which is regarded as a dual operator.

Publications

This manuscript is based on the following articles I have co-authored during my thesis:

— Michell Guzman, Stefan Haar, Salim Perchy, Camilo Rueda, and Frank D. Valencia. Belief, knowledge, lies and other utterances in an algebra for space and extrusion. *Accepted to the Journal of Logical and Algebraic Methods in Programming, JLAMP*, 86, 2016.

— Stefan Haar, Salim Perchy, Camilo Rueda, and Frank D. Valencia. An algebraic view of space/belief and extrusion/utterance for concurrency/epistemic logic. In *Proceedings of the 17th ACM SIGPLAN International Symposium on Principles and Practice of Declarative Programming, PPDP 2015*, pages 161–172, 2015.

— Michell Guzman, Salim Perchy, Camilo Rueda, and Frank D. Valencia. Deriving inverse operators for modal logics. In *Accepted to the 13th International Colloquium on Theoretical Aspects of Computing, ICTAC 2016*, 2016.

— Salim Perchy and Frank D. Valencia. Opinions and beliefs as constraint system operators. In *Technical Communications of the 31st International Conference on Logic Programming, ICLP 2015*, 2015.

Outline of this Work

After this introduction the document follows with Chapter 2 covering the necessary background on order theory to properly introduce constraint systems. This in turn allows to cover spatial constraint systems which are the main concept used

in our theory of mobility in declarative programming. Chapter 3 defines extrusion functions as the main tool to represent mobility of information in a distributed system, it also introduces derived notions to increase the expressiveness of our paradigm. Moreover, the problem of deriving extrusion functions for pre-existing configurations of distributed systems is also considered, and in addition to this, different properties pertinent to the existence of extrusion functions in constraint systems are studied.

The rest of the thesis is dedicated to illustrate the use of our constraint systems for different scenarios where one can interpret mobility of information as a main contextual feature. After a brief background on modal logics, Chapter 4 introduces Kripke constraint systems which are central to modeling epistemic concepts in our structures. Subsequently, a logic of belief and utterance is specified and later characterized by a semantical interpretation using an extension to the Kripke constraint system. This extension is devised by identifying the proper structures to use for introducing extrusion functions as studied in the previous chapter. Chapter 5 provides a representation of knowledge in terms of a derived operator in the constraint systems that specifies global information. Later in the chapter, it is shown that this interpretation is complete with respect to the properties of S4 knowledge.

Finally, Chapter 6 is concerned with expressing time specification by using Time constraint systems. After defining this new type of constraint system, a fragment of Linear Temporal Logic is semantically characterized using its operators and furthermore, properties of the past and future operators of LTL are proved valid using the semantics. The thesis is brought to an end in Chapter 7 where concluding remarks and related work concerning the study and/or representation of epistemic and time concepts are given. Future work pertaining how all these related approaches can be encoded in our formalisms is also written out.

For convenience, we included an index table with the different notation symbols and definitions used throughout the document. Also, in Chapter 7 we provided a summary table of the relationship between constraint elements and operators and those from the different interpretation of space and extrusion.

Chapter 2

Background on Constraint Systems

In this chapter we cover constraint systems, the foundational work that we build upon in this thesis. A constraint system is an algebraic structure where its elements represent partial information. We provide some background on domain theory followed by the introduction of flat constraint systems. Afterwards, we define spatial constraint systems; an extension introducing the concepts of agent and space.

2.1 Order Theory

This section gives a general background on order theory and especially lattice theory. We describe in a terse manner the concepts used throughout this chapter, thus the reader is referred to [DP02, Bly06, GHK+03, AJ94] for an in-depth introduction to these topics.

Posets and Maps

Definition 2.1.1 (Poset)**.** *Given a set of elements P and a binary relation $\sqsubseteq$ on P, we define a partially ordered set $(P, \sqsubseteq)$ where for all $a, b, c \in P$ we have;*

i. $a \sqsubseteq a$ *(reflexivity),*

ii. $a \sqsubseteq b$ *and* $b \sqsubseteq a$ *imply* $a = b$ *(antisymmetry),*

iii. if $a \sqsubseteq b$ *and* $b \sqsubseteq c$ *then* $a \sqsubseteq c$ *(transitivity).*

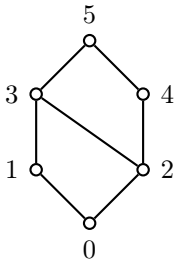


Figure 2.1 – Hasse diagram of a poset

The inverse of the ordering relation is denoted by the symbol $\sqsupseteq$. We write $a \sqsubset b$ if $a \sqsubseteq b$ but $a \neq b$ and $a \not\sqsubseteq b$ when $a \sqsubseteq b$ is false (we define $\sqsupset$ and $\not\sqsupseteq$ dually). It might be the case that two elements are not related (i.e. $a \not\sqsubseteq b$ and $b \not\sqsubseteq a$), we denote this by writing $a \parallel b$.

We use a graphical representation of posets called the *Hasse diagram* wherein elements of the poset are represented as circles. An element is related to another if there a line is drawn connecting its circle with another one above it (see Figure 2.1 as an example).

Example 2.1.1 (Powerset)**.** *Let S be any set. We construct a poset where the elements are all the subsets of S and the ordering relation is set inclusion, i.e. $(\mathcal{P}(S), \subseteq)$. Moreover, $(\mathcal{P}(S), \supseteq)$ is also a poset.*

A frequently used type of subset of elements in a poset is the *directed set*, its definition follows.

Definition 2.1.2 (Directed/Filtered set)**.** *Given a poset $(P, \sqsubseteq)$, we say that $D \subseteq P$ is an* upward directed set *(resp.* downward directed set*) if for every $a, b \in D$ there exists $c \in D$ s.t. $a \sqsubseteq c$ and $b \sqsubseteq c$ (resp. $c \sqsubseteq a$ and $c \sqsubseteq b$).*

Upward/downward directed sets are also referred to as just *directed/filtered sets*. We continue with some definitions for maps between posets.

Definition 2.1.3 (Maps)**.** *Let P and Q be posets. A map $f : P \to Q$ is said to be:*

i. ***order-preserving*** *(also called* ***monotone****) if $a \sqsubseteq b$ in P implies $f(a) \sqsubseteq f(b)$ in Q,*

ii. ***order-reflecting*** *if* $f(a) \sqsubseteq f(b)$ *in* Q *implies* $a \sqsubseteq b$ *in* P*,*

iii. ***order-embedding*** *if it is order-preserving and order-reflecting,*

iv. ***order-isomorphism*** *if it is an order-embedding and surjective,*

v. ***self-map*** *if* $P = Q$*.*

Lastly, a particular correspondence in order theory between two maps on posets is that of a *galois connection.*

Definition 2.1.4 (Galois connection)**.** *Given two posets* $(P, \sqsubseteq)$ *and* $(Q, \sqsubseteq)$ *and two maps* $f : P \to Q$ *and* $g : Q \to P$*, the pair* (f, g) *form a* [monotone] Galois connection *if for all* $a \in P$ *and* $b \in Q$ *we have;*

$$f(a) \sqsubseteq b \text{ iff } a \sqsubseteq g(b).$$

Lattices

Before introducing lattices, it is necessary to define the concept of *bounds.*

Definition 2.1.5 (Bounds)**.** *An element* $c \in P$ *is said to be an* upper bound *(resp.* lower bound*) of subset* S *in poset* $(P, \sqsubseteq)$ *if for all* $a \in S$ *we have that* $a \sqsubseteq c$ *(resp.* $c \sqsubseteq a$*).*

We name the sets of all possible lower and upper bounds as S^l and S^u respectively. They can be calculated as follows:

$$S^l = \{c \in P \mid (\forall a \in S)\, a \sqsupseteq c\} \tag{2.1.1}$$

$$S^u = \{c \in P \mid (\forall a \in S)\, a \sqsubseteq c\} \tag{2.1.2}$$

We call the minimal element of S^u the *least upper bound* (lub) of S and the maximal element of S^l the *greatest lower bound* (glb) of S. When they exist, these two elements are unique because $\sqsubseteq$ is antisymmetric (see Definition 2.1.1).

The least upper bound (lub) is also sometimes referred to as *supremum* or *join* and the greatest lower bound (glb) is also referred to as *infimum* or *meet.*

Notation 2.1.1. *We shall write* $\bigsqcup S$ *and* $\bigsqcap S$ *to denote the join and the meet of the set* S *respectively. Alternatively, we write* $c \sqcup d$ *and* $c \sqcap d$ *to denote the lub and glb of the set* $\{c, d\}$ *respectively.*

We now continue with the definition of a lattice.

Definition 2.1.6 (Lattice)**.** *Let* $L = (P, \sqsubseteq)$ *be a poset where* $c \sqcup d$ *and* $c \sqcap d$ *exist for all* $c, d \in P$*, then* L *is also called a* Lattice*.*

A lattice L can also be seen as an algebraic structure with operations for calculating bounds, i.e. $L = (P, \sqsubseteq, \sqcup, \sqcap)$. We proceed to introduce special important types of lattices.

Definition 2.1.7 (Complete lattice)**.** *A lattice* L *is a* Complete Lattice *if* $\bigsqcup S$ *and* $\bigsqcap S$ *exist for every subset* S *of the elements of the* L*.*

Remark 2.1.1. *If a lattice* L *is complete, by definition there exist elements* $\top = \bigsqcup S$ *and* $\bot = \bigsqcap S$ *when* S *is the set of all elements or the empty set. These two elements are called* top *(or global supremum) and* bottom *(or global infimum) respectively. They are usually specified when denoting the lattice as an algebraic structure, i.e.* $L = (P, \sqsubseteq, \sqcup, \sqcap, \top, \bot)$*.*

Example 2.1.2. *Let us consider again Figure 2.1. We calculate* $2^u = \{3, 4, 5\}$ *and* $1^u = \{3, 5\}$*, therefore* $\{1, 2\}^u = \{3, 5\}$ *and* $1 \sqcup 2 = 3$ *because* $3 \sqsubseteq 5$*. In the same way we can calculate* $3 \sqcap 4 = 2$*. Let* S *be the set of all elements of the lattice, then* $\bigsqcup S = 5 = \top$ *and* $\bigsqcap S = 0 = \bot$

Example 2.1.3 (Powerset lattice)**.** *Let* S *be any set. We construct the powerset lattice of* S *as* $L = (\mathcal{P}(S), \subseteq, \cup, \cap, \emptyset, S)$*. Moreover,* $L' = (\mathcal{P}(S), \supseteq, \cap, \cup, S, \emptyset)$ *is called the reverse powerset lattice of* S*.*

Both L and L' in Example 2.1.3 are complete lattices. Some important properties of the join and meet operators need to be introduced.

Theorem 2.1.1. *Let* L *be a lattice. For all* $a, b, c \in L$ *we have that:*

— $(a \sqcup b) \sqcup c = a \sqcup (b \sqcup c)$ *and* $(a \sqcap b) \sqcap c = a \sqcap (b \sqcap c)$

— $a \sqcup b = b \sqcup a$ *and* $a \sqcap b = b \sqcap a$

— $a \sqcup a = a$ *and* $a \sqcap a = a$

— $a \sqcup (a \sqcap b) = a$ *and* $a \sqcap (a \sqcup b) = a$

We characterize the elements of a lattice that capture the notion of finiteness.

Definition 2.1.8 (Compact elements)**.** *Given a lattice L and an element $k \in L$, we say that k is compact if for every subset S of L $k \sqsubseteq \bigsqcup S$ implies $k \sqsubseteq \bigsqcup T$ for some finite $T \subseteq S$.*

The set of compact elements of a lattice L is denoted by $K(L)$.

Example 2.1.4. *Let L be the powerset lattice of S, the set $K(L)$ are all the finite subsets of S. Moreover, let L' be a reverse power lattice of S, the set $K(L)$ are all the co-finite subsets of S. Recall that a set P is cofinite if its complement is finite (i.e. $P \backslash S$ is finite).*

Definition 2.1.9 (Algebraic lattice)**.** *A complete lattice L is said to be* algebraic *if for each $c \in L$ we have;*

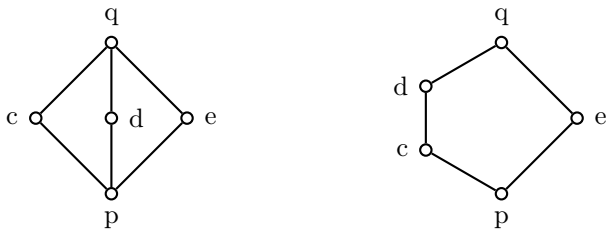
$$c = \bigsqcup \{\ k \in K(L) \mid k \sqsubseteq c\ \}.$$

Intuitively, an algebraic lattice is a lattice where each element can be approximated by the finite elements below it. It can easily be shown that lattices in Example 2.1.3 are algebraic. We continue with yet another important type of lattice.

Definition 2.1.10 (Distributive lattice)**.** *A lattice L is said to be distributive if for every $a, b, c \in L$ we have;*

$$a \ \sqcup\ (b \ \sqcap\ c) = (a \ \sqcup\ b) \ \sqcap\ (a \ \sqcup\ c)$$

Checking distributivity can be cumbersome, specially when there is a significant number of elements in the lattice. We describe another way for verifying this property.



(a) $\mathbf{M}_3$ (b) $\mathbf{N}_5$

Figure 2.2 – The $\mathbf{M}_3$ and $\mathbf{N}_5$ lattices, both are non-distributive

Definition 2.1.11 (Sub-lattice)**.** *Given $P \subset Q$, a lattice $M = (P, \sqsubseteq, \sqcup, \sqcap)$ is a sub-lattice of lattice $L = (Q, \dot{\sqsubseteq}, \dot{\sqcup}, \dot{\sqcap})$ (written as $M \rightarrowtail L$) if for each $c, d \in P$ then $c \mathbin{\dot{\sqcup}} d \in P$ and $c \mathbin{\dot{\sqcap}} d \in P$.*

Theorem 2.1.2 ($\mathbf{M}_3$-$\mathbf{N}_5$ theorem)**.** *A lattice L is non-distributive if and only if $\boldsymbol{M}_3 \rightarrowtail L$ or $\boldsymbol{N}_5 \rightarrowtail L$.*

Proof. See [DP02]. □

Lastly, another significant type of lattice is that of a *frame*, also called a *complete Heyting Algebra* (cHa).

Definition 2.1.12 (Frame)**.** *A lattice L is a* frame *if it is complete and it satisfies the next equation:*

$$c \sqcup \bigsqcap S = \bigsqcap \{c \sqcup s \mid s \in S\} \tag{2.1.3}$$

By definition, a frame is also a distributive lattice (Definition 2.1.10). We finish our exposition of lattices with a mathematical structure that serves as a semantic interpretation of boolean logics.

Definition 2.1.13 (Boolean algebra)**.** *A* boolean algebra *is a structure $(B, \wedge, \vee, \prime, 0, 1)$ where:*

i. $(B, \wedge, \vee, 0, 1)$ *is a distributive lattice,*

ii. $a \wedge 1 = a$ *and* $a \vee 0 = a$ *for all* $a \in B$*,*

iii. $a \wedge a' = 0$ *and* $a \vee a' = 1$ *for all* $a \in B$*.*

Example 2.1.5. *The powerset lattice of a set S of elements can form a boolean algebra where $B = S$, $\wedge = \cup$, $\vee = \cap$, $\prime = S\backslash\cdot$, $0 = \emptyset$, $1 = S$. Moreover, the reverse power lattice of S forms the boolean algebra $(S, \cap, \cup, S\backslash\cdot, 0 = S, 1 = \emptyset)$.*

An important property of maps with respect to the calculation of bounds is completeness. We formalize this concept as follows.

Definition 2.1.14. *Given two lattices L and L', a set $S \in L$ and a map $f : L \to L'$ we say:*

i. if $f(\bigsqcup S) = \bigsqcup f(S)$ then;

a. f is [upward]-continuous (or Scott-continuous) if S is directed *(see Def. 2.1.2),*

b. f is join-complete if S is any set.

ii. if $f(\bigsqcap S) = \bigsqcap f(S)$ then;

a. f is downward-continuous,

b. f if meet-complete if S is any set.

2.2 Constraint Systems

We proceed to explain the notion of flat constraint system and the more recent notion of spatial constraint system [KPPV12]. Following [BDPP95] we formalize constraint systems as *complete algebraic lattices* (an alternative syntactic characterization of cs, akin to Scott information systems, is given in [SRP91, PSSS93]).

The elements of the lattice, the *constraints*, represent (partial) information. A constraint c can be viewed as an *assertion* (or a *proposition*). The lattice order $\sqsubseteq$ is meant to capture entailment of information: $c \sqsubseteq d$, alternatively written $d \sqsupseteq c$, means that the assertion d represents at least as much information as c. Thus we may think of $c \sqsubseteq d$ as saying that d *entails* c or that c can be *derived* from d. The *least upper bound (lub)* operator $\sqcup$ represents join of information; $c \sqcup d$, the least element in the underlying lattice above c and d. Thus $c \sqcup d$ can be seen as an assertion stating that both c and d hold. The top element represents the lub of all, possibly inconsistent, information, hence it is referred to as *false*. The bottom element *true* represents the empty information.

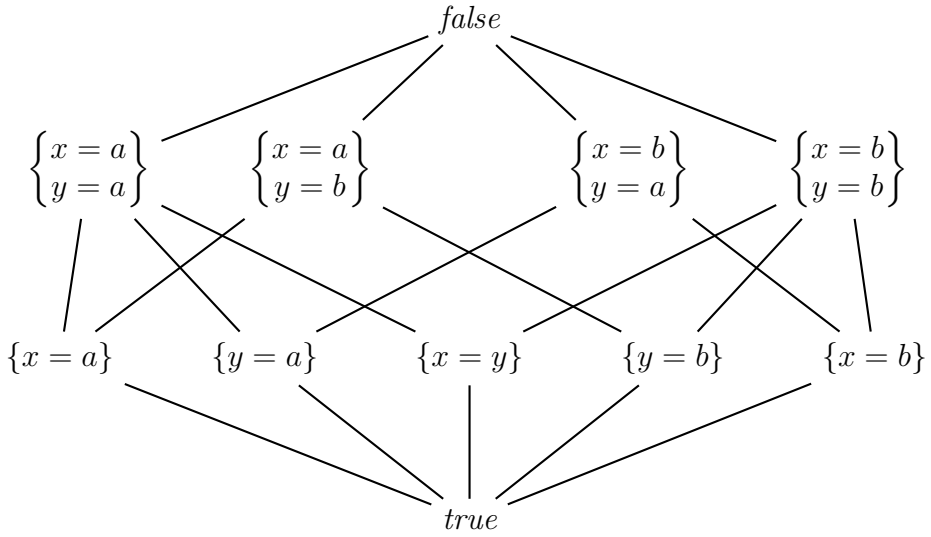


Figure 2.3 – A Herbrand constraint system

Definition 2.2.1 (Constraint Systems [BDPP95])**.** *A constraint system (cs)* $\mathcal{C}$ *is a complete algebraic lattice* $(Con, \sqsubseteq)$*. The elements of Con are called* constraints*. The symbols* $\sqcup$*, true and false will be used to denote the least upper bound (lub) operation, the bottom, and the top element of* $\mathcal{C}$*, respectively.*

The lattice representation of information in constraint systems is reminiscent of the algebraic presentation of *geometric logic* [Vic96]. Next, we describe two typical concrete constraint systems.

Example 2.2.1 (Herbrand Constraint System [BDPP95, SRP91])**.** *The Herbrand cs captures* syntactic *equality between terms* $t, t', \ldots$ *built from a first-order alphabet* $\mathcal{L}$ *with variables* $x, y, \ldots$*, function symbols, and equality* $=$*. The constraints are (equivalent classes of) sets of equalities over the terms of* $\mathcal{L}$*: E.g.,* $\{x = t, y = t\}$ *is a constraint. The relation* $c \sqsubseteq d$ *holds if the equalities in c follow from those in d: E.g.,* $\{x = y\} \sqsubseteq \{x = t, y = t\}$*. The constraint false is the set of all term equalities in* $\mathcal{L}$ *and true is (the equivalence class of) the empty set. The compact elements are the (equivalence class of) finite sets of equalities. The lub is the (equivalence class of) set union. Figure 2.3 is the hasse diagram of a Herbrand cs with variables* $\{x, y\}$ *and constants* $\{a, b\}$ *with* $a \neq b$*.* □

In the above example constraints are represented as sets of equations and thus the join (lub) of constraints corresponds to the equivalence class of the *union* of their equations. We can also view a constraint c as a representation of a set of variable assignments [ABP+11]. For instance a constraint $x > 42$ can be thought of as the set of assignments mapping x to a value greater than 42; i.e., the solutions to (or models of) $x > 42$. In this case the join of constraints naturally corresponds to the *intersection* of their assignments, *false* as the empty set of assignments, and *true* as the set of all assignments.

Example 2.2.2 (Boolean Constraint System [ABP+11])**.** *Let Φ be a set of* primitive propositions*. A boolean (or truth) assignment π over Φ is a total map from Φ to the set $\{0, 1\}$. We use $\mathcal{A}(\Phi)$ to denote the set of all such boolean assignments. We can now define the boolean cs $\mathbf{B}(\Phi)$ as $(\mathcal{P}(\mathcal{A}(\Phi)), \supseteq)$: The powerset of assignments ordered by $\supseteq$. Thus constraints in Con are sets of assignments, $\sqsubseteq$ is $\supseteq$, false is $\emptyset$, true is $\mathcal{A}(\Phi)$, the join operator $\sqcup$ is $\cap$, and the meet operator $\sqcap$ is $\cup$. A constraint c in $\mathbf{B}(\Phi)$ is compact iff $\mathcal{A}(\Phi) \setminus c$ is a finite set.* □

Notice that logic propositions can be straightforwardly interpreted as constraints in $\mathbf{B}(\Phi)$. Let $\mathcal{L}_0(\Phi)$ be the propositional language built from Φ by the grammar

$$\phi, \psi, \ldots \;:=\; p \mid \phi \wedge \psi \mid \neg\phi \tag{2.2.1}$$

where $p \in \Phi$. We shall use the classical abbreviations $\phi \vee \psi$ for $\neg(\neg\phi \wedge \neg\psi)$, $\phi \Rightarrow \psi$ for $\neg\phi \vee \psi$, F for $p \wedge \neg p$, and T for $\neg$F. A boolean assignment π *satisfies* ϕ iff $\pi \models \phi$ where $\models$ is defined inductively as follows: $\pi \models p$ iff $\pi(p) = 1$, $\pi \models \phi \wedge \psi$ iff $\pi \models \phi$ and $\pi \models \psi$, and $\pi \models \neg\phi$ iff $\pi \not\models \phi$. We interpret each formula ϕ as the constraint $\mathbf{B}[\![\phi]\!] \stackrel{\text{def}}{=} \{\pi \in \mathcal{A}(\Phi) \mid \pi \models \phi\}$ in $\mathbf{B}(\Phi)$. Clearly $\mathbf{B}[\![\phi]\!] \sqsubseteq \mathbf{B}[\![\psi]\!]$ holds iff $\psi \Rightarrow \phi$ is valid, i.e., satisfied by every truth assignment.

Remark 2.2.1 (Boolean Implication)**.** *Constraint systems of the form $(\mathcal{P}(U) \supseteq)$ as $\mathbf{B}(\Phi)$ in Example 2.2.2, are standard examples of a Boolean algebra [GHK+03]. Given the constraints $c, d \in \mathcal{P}(U)$, the negation constraint $\neg c$ and the implication constraint $c \Rightarrow d$ in $\mathcal{P}(U)$ are defined as $U \setminus c$ and $\neg c \;\cup\; d$, respectively.*

Other typical examples include constraint system for streams (the Kahn cs), rational intervals, and first-order theories [SRP91].

Spatial Constraint Systems

The authors of [KPPV12] extended the notion of cs to account for distributed and multi-agent scenarios where agents have their own space for their local information and performing their computations.

Locality and Nested Spaces

Intuitively, each agent i has a *space* function $[\cdot]_i$ from constraints to constraints. Recall that constraints can be viewed as assertions. We can then think of

$$[c]_i \tag{2.2.2}$$

as an assertion stating that c is a piece of information that resides *within a space attributed to agent* i. An alternative *epistemic interpretation* of $[c]_i$ is an assertion stating that agent i *believes* c or that c holds within the space of agent i (but it may or may not hold elsewhere). Both interpretations convey the idea that c is local to agent i.

Following the above intuition, the assertion

$$[[c]_j]_i \tag{2.2.3}$$

is a hierarchical spatial specification stating that c holds within the local space the agent i attributes to agent j. Nesting of spaces such as in $[[\ldots[c]_{i_m}\ldots]_{i_2}]_{i_1}$ can be of any depth.

Parallel Spaces

We can think of a constraint of the form

$$[c]_i \sqcup [d]_j \tag{2.2.4}$$

as an assertion specifying that c and d hold within two *parallel/neighboring* spaces that belong to agents i and j, respectively. From a computational/ concurrency point of view, we think of $\sqcup$ as parallel composition. As mentioned before, from a logic point of view the join of information $\sqcup$ corresponds to conjunction.

We can combine the above parallel and hierarchical specifications to express more complex spatially distributed multi-agent systems. Consider for example

$$[a \sqcup [b]_i \sqcup [c]_j]_i \sqcup [d]_j$$

where agent i has a space within his own space, and agent j has two spaces one in parallel with the outer space of agent i, and other inside it.

An n-agent *spatial constraint system (n-scs)* is a cs parametric in n self-maps $[\cdot]_1, \ldots, [\cdot]_n$ capturing the above intuitions.

Definition 2.2.2 (Spatial Constraint System [KPPV12])**.** *An n-agent* spatial constraint system (n-scs) $\boldsymbol{C}$ *is a cs* $(Con, \sqsubseteq)$ *equipped with n self-maps $[\cdot]_1, \ldots, [\cdot]_n$ over its set of constraints Con such that for each function $[\cdot]_i : Con \to Con$:*

S.1 $[true]_i = true$*, and*

S.2 $[c \sqcup d]_i = [c]_i \sqcup [d]_i$ *for each $c, d \in Con$.*

Henceforth, given an n-scs $\mathbf{C}$, we refer to each $[\cdot]_i$ as the *space* (or space function) of the agent i in $\mathbf{C}$. We use $(Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$ to denote the corresponding n-scs with space functions $[\cdot]_1, \ldots, [\cdot]_n$. We shall often omit components of an n-scs tuple when they are unnecessary or clear from the context. We shall simply write scs when n is unimportant.

We now give some intuition about the space properties. Property S.1 in Definition 2.2.2 requires space functions to be strict maps (i.e bottom preserving). Intuitively, it states that having an empty local space amounts to nothing. Property S.2 states that space functions preserve (finite) lubs and it allows us to join and distribute the local information of agent i.

Remark 2.2.2 (Monotone Spaces)**.** *Notice that S.2 implies that space function are* order-preserving *(or monotone): i.e., if $c \sqsubseteq d$ then $[c]_i \sqsubseteq [d]_i$. Intuitively, if c can be derived from d then any agent i should be able to derive c from d within its own space.*

Proof. Assume $c \sqsubseteq d$, thus $d = c \sqcup d$. Then $[d]_i = [c \sqcup d]_i$. Using S.2 we have $[d]_i = [c]_i \sqcup [d]_i$, hence $[c]_i \sqsubseteq [d]_i$. □

Shared and Global Information

Some noteworthy derived spatial constructions are shared-spaces and globality.

Definition 2.2.3 (Global Information)**.** *Let $\mathcal{C}$ be an n-scs with space functions $[\cdot]_1, \ldots, [\cdot]_n$ and G be a non-empty subset of $\{1, \ldots, n\}$.* Group-spaces $[\cdot]_G$ *and* global *information $[\![\cdot]\!]_G$ of G in $\mathcal{C}$ are defined as:*

$$[c]_G \stackrel{\texttt{def}}{=} \bigsqcup_{i \in G} [c]_i \quad \textit{and} \quad [\![c]\!]_G \stackrel{\texttt{def}}{=} \bigsqcup_{j=0}^{\infty} [c]_G^j \tag{2.2.5}$$

where $[c]_G^0 \stackrel{\texttt{def}}{=} c$ and $[c]_G^{k+1} \stackrel{\texttt{def}}{=} [[c]_G^k]_G$.

The constraint $[c]_G$ means that c holds in the spaces of agents in G. The constraint $[\![c]\!]_G$ entails $[[\ldots[c]_{i_m}\ldots]_{i_2}]_{i_1}$ for any $i_1, i_2, \ldots, i_m \in G$. Thus it realizes the intuition that c holds *globally* wrt G: c holds in each nested space involving only the agents in G. In particular if G is the set of all agents, $[\![c]\!]_G$ means that c holds *everywhere*. From the epistemic point of view $[\![c]\!]_G$ is related to the notion of common-knowledge of c [FHMV95].

2.3 Summary

We gave a terse introduction to lattices, the main stepping stone from order theory we are going to use throughout the thesis. We introduced various types of lattices, their properties, maps between them and proceeded to define constraint systems as complete algebraic lattices representing partial information. More in-depth texts on these subjects are [DP02, SRP91].

Spatial constraint systems, an important structure from which our worked is based are also introduced here. The next chapter we introduce the concept of extrusion w.r.t. space in constraint systems, we again use maps between lattices as the mathematical concept to achieve this.

Chapter 3

Spatial Constraint Systems with Extrusion

We shall now introduce a new notion, that of *spatial constraint systems with extrusion (scse)* and use it to specify simple examples of mobile and epistemic behavior. We also investigate the problem of extending any given arbitrary spatial constraint system to a scse. We will then state some distinctive properties of space and extrusion as well as correspondences between these two concepts that will be used throughout the following chapters.

3.1 Extrusion as the right inverse of Space

In spatially distributed systems an agent can transfer information from its space to the outside. We shall refer to this kind of transmission as *extrusion*. The extruded information is posted outside, possibly addressed to some other agent. Our epistemic view of extrusion is what we shall call *utterance*. An agent may utter information which will then be available to others. The uttered information may be inconsistent with the agent's own beliefs, in particular it could be a *lie*.

Let us now extend spatial constraint systems with extrusion. First recall that given a function $f : X \to Y$, we say that $g : Y \to X$ is a *right inverse* (or *section*) of f iff $f(g(y)) = y$ for every $y \in Y$. Similarly, given $g : Y \to X$ we say that $f : X \to Y$ is a *left inverse* (or *retraction*) of g iff $f(g(y)) = y$ for every $y \in Y$.

We shall equip each agent i with an *extrusion* function $\uparrow_i : Con \to Con$. Intuitively, within a space context $[\cdot]_i$, the assertion $\uparrow_i c$ specifies that c must be posted outside of (or extruded from) agent $i's$ space. This will be captured by requiring the *extrusion* property:

$$\text{E.1: } [\uparrow_i c]_i = c. \tag{3.1.1}$$

In other words, we view *extrusion/utterance* as the right inverse of *space/belief* (and thus space/belief as the left inverse of extrusion/utterance).

A *spatial constraint systems with extrusion (scse)* is an scs with right inverses for each one of its space functions.

Definition 3.1.1 (Spatial Constraint System with Extrusion)**.** *An n-agent* spatial constraint system with extrusion (n-scse) $\boldsymbol{C}$ *is an n-scs $(Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$ equipped with n self-maps $\uparrow_1, \ldots, \uparrow_n$ over Con such that $\uparrow_i$ is a right inverse of $[\cdot]_i$. More precisely, each self-map $\uparrow_i$ of $\boldsymbol{C}$ satisfies the following condition:*

E.1 $[\uparrow_i c]_i = c$ *for every $c \in Con$.*

Henceforward we shall refer to each $\uparrow_i$ as the *extrusion* function of agent i in $\mathbf{C}$. We use $(Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n, \uparrow_1, \ldots, \uparrow_n)$ to denote the corresponding n-scs $(Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$ with extrusion functions $\uparrow_1, \ldots, \uparrow_n$.

We shall study additional properties (i.e., axioms) for extrusion in Section 3.4. In the next chapter we show that E.1 already allows us to specify meaningful spatial and epistemic behavior.

3.2 Derived Notions and Applications

We now introduce some derived general constructs to illustrate the expressiveness of extrusion. First, we need a general notion of implication.

Heyting Implication

In Remark 2.2.1 (Section 2.2), we discussed an interpretation of implication that works for the Boolean cs. We can define a general form of implication by

adapting the corresponding notion from Heyting Algebras [Vic96] to constraint systems.

Intuitively, a *Heyting implication* $c \to d$ in our settings corresponds to the *weakest constraint* one needs to join c with to derive d: The greatest lower bound $\bigsqcap\{e \mid e \sqcup c \sqsupseteq d\}$. Similarly, the negation of a constraint c, written $\sim c$, can be seen as the *weakest constraint inconsistent* with c, i.e., the greatest lower bound $\bigsqcap\{e \mid e \sqcup c \sqsupseteq \mathit{false}\} = c \to \mathit{false}$.

Definition 3.2.1 (Heyting Implication and Negation)**.** *Let $\mathcal{C}$ be a constraint system $(Con, \sqsubseteq)$. Define $c \to d$ as:*

$$\bigsqcap\{e \mid e \sqcup c \sqsupseteq d\} \tag{3.2.1}$$

and $\sim c$ as $c \to \mathit{false}$.

The above construction corresponds to (intuitionistic) implication in lattices that are *frames* (see Definition 2.1.12) [Vic96].

Remark 3.2.1. *The Boolean constraint system (Example 2.2.2) is a frame since meets are unions and joins are intersections so the distributive requirement is satisfied. Furthermore, for cs's of the form $(\mathcal{P}(U), \supseteq)$, as e.g., $\mathbf{B}(\Phi)$, the operators $\to$ and $\sim$ are known to coincide with the constructions $\Rightarrow$ and $\neg$ defined in Remark 2.2.1, since boolean implication and boolean negation are particular cases of Heyting implication and Heyting negation [Vic96].*

A typical example of a standard constraint system that is not a frame is Herbrand's. To see this consider the Herbrand constraint system in Figure 2.3 and consider the constraints $c = \{x = a\}, d = \{x = b\}$ and $e = \{x = a, y = a\}$. We have:

$$c \sqcup (d \sqcap e) \neq (c \sqcup d) \sqcap (c \sqcup e)$$
$$c \sqcup \mathit{true} \neq \mathit{false} \sqcap e$$
$$c \neq e$$

For a more intuitive and graphic way to prove this we use Theorem 2.1.2. Let L (Figure 3.1) be a lattice constructed out of the elements *true*, *false* and the

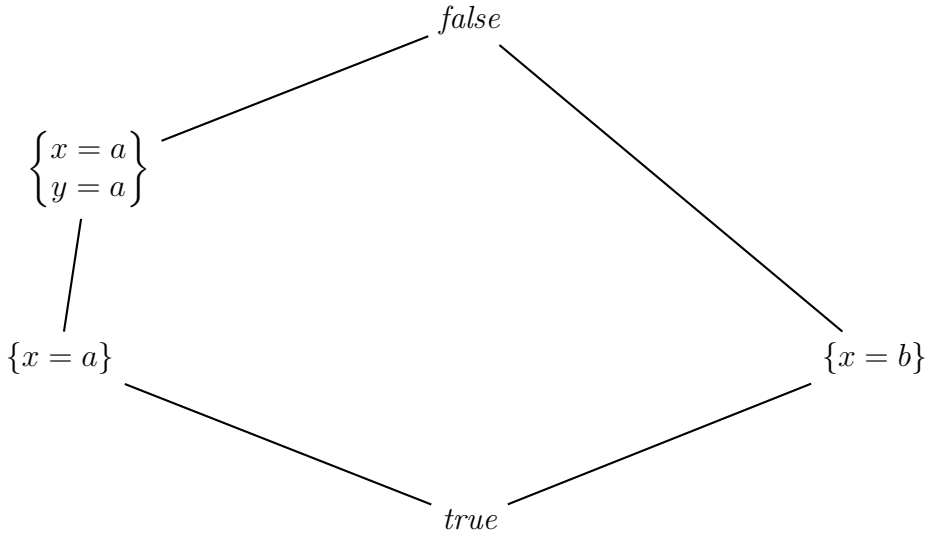


Figure 3.1 – The Herbrand sub-lattice $\mathbf{N}_5$

above-mentioned c, d and e from the Herbrand cs $\mathbf{H}$ from Figure 2.3 (we keep the same ordering relation). It is then easy to show that $L \rightarrowtail \mathbf{H}$ (i.e. L is a sub-lattice of $\mathbf{H}$) and that L is isomorphic to $\mathbf{N}_5$, thus proving that $\mathbf{H}$ is not distributive nor a frame.

The main property of Heyting implication we shall use in our applications is a form of modus ponens.

Lemma 3.2.1 (Modus-Ponens)**.** *Suppose that* $(Con, \sqsubseteq)$ *is a frame. Then for every* c, d*, we have:*

$$c \sqcup (c \to d) = c \sqcup d \tag{3.2.2}$$

Proof. We need to prove $c \sqcup \bigsqcap\{e \mid d \sqsubseteq e \sqcup c\} = c \sqcup d$. Recall that by definition joins distribute over arbitrary meets in any frame.

— First we prove $c \sqcup \bigsqcap\{e \mid d \sqsubseteq e \sqcup c\} \sqsubseteq c \sqcup d$. Let $S = \{e \mid d \sqsubseteq e \sqcup c\}$. Since $d \in S$, we conclude $\bigsqcap S \sqsubseteq d$. Thus, $c \sqcup \bigsqcap S \sqsubseteq c \sqcup d$ as wanted.

— We now prove $c \sqcup d \sqsubseteq c \sqcup \bigsqcap\{e \mid d \sqsubseteq e \sqcup c\}$. Let $S = \{e \mid d \sqsubseteq e \sqcup c\}$. Distributing the join over the meet we obtain $c \sqcup \bigsqcap S = \bigsqcap\{c \sqcup e \mid e \in S\}$. Since each $c \sqcup e \sqsupseteq d$ for every $e \in S$ then $d \sqsubseteq \bigsqcap\{c \sqcup e \mid e \in S\} = c \sqcup \bigsqcap S$. Therefore $c \sqcup d \sqsubseteq c \sqcup c \sqcup \bigsqcap S = c \sqcup \bigsqcap S$.

□

Another basic but important property of the Heyting implication is its direct link to the ordering relation of the constraint system, we formalize it in the next lemma.

Lemma 3.2.2. *Suppose that $(Con, \sqsubseteq)$ is a frame. For every c, d we the following is true:*

$$c \to d = true \quad iff \quad c \sqsupseteq d$$

Proof.

We first prove that $c \sqsupseteq d$ whenever $c \to d = true$. Suppose $c \to d = true$, we join c to both sides to obtain $c \sqcup (c \to d) = c$. Using Modus-Ponens (Lemma 3.2.1) we get $c \sqcup d = c$ and conclude that $c \sqsupseteq d$.

Finally we prove that if $c \sqsupseteq d$ then $c \to d = true$. Recall that $c \to d = \bigsqcap S$ where $S = \{e \mid e \sqcup c \sqsupseteq d\}$. Suppose $c \sqsupseteq d$ then $true \in S$ and $c \to d = \bigsqcap S = true$. □

Heyting implication can be used in combination with our spatial constructions to specify meaningful computational and social behavior.

Remark 3.2.2. *For the applications examples in this chapter, we fix an scse $(Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n, \uparrow_1, \ldots, \uparrow_n)$. Furthermore we assume $(Con, \sqsubseteq)$ is a frame.*

Lying Agents

A lie is not necessarily a false statement but rather a statement that deviates from what its author actually knows, believes or holds to be true [VDVESW12]. Instances of this concept can be realized in our setting by thinking of an (intentional) lie or *hoax* as the uttering/extrusion of a statement by an agent which is *inconsistent* with what he or she believes to be true.

Example 3.2.1 (Hoax)**.** *Suppose that $c \sqcup d = false$. The assertion*

$$[c \sqcup \uparrow_i d]_i \tag{3.2.3}$$

specifies an agent i that believes c and wishes to utter/extrude d. Since c and d are inconsistent and agent i believes c we can regard d as a hoax or an intentional

lie by agent i. It follows from Definition 3.2.1 that by taking $d = \sim c$ we obtain the weakest statement inconsistent with c. In other words $\sim c$ is the weakest/most general lie by agent i wrt his or her belief c.

We can use the spatial axiom S.2 (Definition 2.2.2) followed by the extrusion axiom E.1 (Definition 3.1.1) to obtain the following derivation of $[c]_i \sqcup d$.

$$\begin{aligned} [c \sqcup \uparrow_i d]_i &= [c]_i \sqcup [\uparrow_i d]_i && \textit{(S.2)} \\ &= [c]_i \sqcup d && \textit{(E.1)} \end{aligned}$$

The transformation from Equation 3.2.3 to $[c]_i \sqcup d$ with $c \sqcup d = false$ illustrates the extrusion of (the lie) d by agent i. □

Communicating Agents

Let us now illustrate hoaxes and communication between agents via extrusion. Recall that we think of $[c]_i \sqcup [d]_j$ as an assertion saying that c and d hold within two *parallel* spaces that belong to agents i and j, respectively.

Example 3.2.2 (Communication)**.** *Let us suppose that we have an agent j who would utter d if she thought $\sim c$ was true. This behavior of agent j can be specified as*

$$[\sim c \rightarrow \uparrow_j d]_j. \tag{3.2.4}$$

Furthermore, suppose that we have an agent i who considers c to be true and yet he wishes to communicate the opposite to agent j. The behavior of agent i can be expressed as

$$[c \sqcup \uparrow_i [\sim c]_j]_i. \tag{3.2.5}$$

Notice that the constraint to be extruded from the space of agent i, i.e., $[\sim c]_j$, can be viewed as a message $\sim c$ addressed to agent j.

The expected result, if i communicates his hoax $\sim c$ to j, is that d gets posted to the outermost position. The communication should take place if the agents' spaces are placed in parallel. In fact we put together Equations 3.2.4 and 3.2.5, we derive

the expected result.

$$
\begin{aligned}
&[\sim c \to \uparrow_j d]_j \ \sqcup\ [c \ \sqcup\ \uparrow_i [\sim c]_j]_i \\
&= [\sim c \to \uparrow_j d]_j \ \sqcup\ [c]_i \ \sqcup\ [\uparrow_i [\sim c]_j]_i && \textit{(S.2 on } [\cdot]_i\textit{)} \\
&= [\sim c \to \uparrow_j d]_j \ \sqcup\ [c]_i \ \sqcup\ [\ \sim c]_j && \textit{(E.1 on } [\cdot]_i\textit{)} \\
&= [\sim c \sqcup\ \sim c \to \uparrow_j d]_j \ \sqcup\ [c]_i && \textit{(S.2 on } [\cdot]_j\textit{)} \\
&= [\sim c \ \sqcup\ \uparrow_j d]_j \ \sqcup\ [c]_i && \textit{(Lemma 3.2.1)} \\
&= d \ \sqcup\ [\sim c]_j \ \sqcup\ [c]_i && \textit{(E.1 on } [\cdot]_j\textit{)}
\end{aligned}
$$

□

Process Mobility

From a declarative programming point of view the construct $c \to d$ can be seen as a *program/computational process* that produces d if the guard c holds true. We can then combine this construct with our extrusion to express meaningful mobile behavior of programs.

Example 3.2.3 (Mobility)**.** *Let us consider the following assertion:*

$$[e \ \sqcup\ \uparrow_i (c \to [d]_i)]_i. \tag{3.2.6}$$

Equation 3.2.6 specifies the sending of a process $c \to [d]_i$ outside the space of agent i that already contains e. Once the process is outside, if c holds, it will put d in i's space. Indeed, with the help of S.2, E.1 and Lemma 3.2.1 we can derive $[e \ \sqcup\ d]_i$ from $c \ \sqcup\ [e \ \sqcup\ \uparrow_i (c \to [d]_i)]_i$ as follows:

$$
\begin{aligned}
&c \ \sqcup\ [e \ \sqcup\ \uparrow_i (c \to [d]_i)]_i \\
&= c \ \sqcup\ [e]_i \ \sqcup\ [\uparrow_i (c \to [d]_i)]_i && \textit{(S.2)} \\
&= c \ \sqcup\ [e]_i \ \sqcup\ c \to [d]_i && \textit{(E.1)} \\
&= c \ \sqcup\ [e]_i \ \sqcup\ [d]_i && \textit{(Lemma 3.2.1)} \\
&= c \ \sqcup\ [e \ \sqcup\ d]_i && \textit{(S.2)}
\end{aligned}
$$

The step corresponding to E.1 shows the extrusion of the process $c \to [d]_i$.

For a more involved example of extrusion of implication processes consider

$$[e \ \sqcup \ \uparrow_i [c \to \uparrow_j [d]_i]_j]_i \tag{3.2.7}$$

Intuitively, the implication process $c \to \uparrow_j [d]_i$ is sent from within the space of i to a parallel space that belongs to j. Then if c holds in that parallel space, $[d]_i$ is extruded from $[\cdot]_j$ and thus d is placed in the space of i from where the implication process was sent.

In fact, after multiple applications of E.1, S.2 and Lemma 3.2.1 we obtain the following:

$$[e \ \sqcup \ \uparrow_i [c \to \uparrow_j [d]_i]_j]_i \ \sqcup \ [c]_j \ \ \sqsupseteq \ \ [e \ \sqcup \ d]_i. \tag{3.2.8}$$

(We use $\sqsupseteq$ instead of $=$ to omit some non essential information that would join $[e \ \sqcup \ d]_i$.)

Notice that $c \to \uparrow_j [d]_i$ above can be seen as an intrusive process wrt agent j since it reports to agent i if c holds in $[\cdot]_j$. □

Outermost Extrusion

We now derive constructions that can be used to specify extrusion to the *outermost* position in arbitrary nested spaces.

Definition 3.2.2 (Global Extrusion)**.** *Let* $\mathbf{C}$ *be an n-scse with extrusion functions $\uparrow_1, \ldots, \uparrow_n$ and G be a non-empty subset of $\{1, \ldots, n\}$.* Group-extrusion $\uparrow_G$ *and* global extrusion $\Uparrow_G$ *of G in* $\mathbf{C}$ *are defined as:*

$$\uparrow_G c \stackrel{\texttt{def}}{=} \bigsqcup_{i \in G} \uparrow_i c \ \ \textit{and} \ \ \Uparrow_G c \stackrel{\texttt{def}}{=} \bigsqcup_{j=0}^{\infty} \uparrow_G^j c \tag{3.2.9}$$

where $\uparrow_G^0 c \stackrel{\texttt{def}}{=} c$ and $\uparrow_G^{k+1} c \stackrel{\texttt{def}}{=} \uparrow_G \uparrow_G^k c$.

Recall the notion of shared space in Definition 2.2.3. The group extrusion $\uparrow_G c$ extrudes c from any space or shared-space of the agents in G. In fact, for any G,

$$[\uparrow_G c]_G \sqsupseteq c \text{ and } \ [\uparrow_G c]_j \sqsupseteq c$$

for any $j \in G$.

Global extrusion $\Uparrow_G c$ can pull c into the outermost position regardless of the nesting depth (of spaces involving the agents in G). One can verify that

$$[[\ldots[\Uparrow_G c\ldots]_{i_m}\ldots]_{i_2}]_{i_1} \sqsupseteq c$$

for every $i_1, i_2, \ldots, i_m \in G$.

Spatial Safety

We conclude this section by combining all our previously derived constructions to specify the extrusion of d to the outermost position if c is present somewhere in a given constraint e with arbitrary nested spaces (e.g. $e = [[a]_j]_i \sqcup [[c]_i]_j$). If c represents an *undesired* event in e then d can be used as a witness of its presence.

Example 3.2.4 (Spatial Search)**.** *Suppose that G is the set of all agents. The assertion $c \to \Uparrow_G d$ specifies that d will be extruded to the outermost position if c holds. We can use the global space construction $[\![c \to \Uparrow_G d]\!]_G$ in Definition 2.2.3 to specify that $c \to \Uparrow_G d$ is everywhere.*

We can verify that for any spatial constraint e where c holds somewhere, i.e., for any e such that

$$e \sqsupseteq [[\ldots[c]_{i_m}\ldots]_{i_2}]_{i_1} \tag{3.2.10}$$

for some $i_1, i_2, \ldots, i_m \in G$, we have

$$e \sqcup [\![c \to \Uparrow_G d]\!]_G \sqsupseteq d. \tag{3.2.11}$$

□

3.3 Limit Preservation

In the following sections we will often refer to preservation of some limits by space functions. Let $\mathbf{C}$ be an scs with constraints *Con*. A space function $[\cdot]_i$ of $\mathbf{C}$ *preserves* the supremum of a set $S \subseteq Con$ iff $[\bigsqcup S]_i = \bigsqcup\{[c]_i \mid c \in S\}$. The preservation of the infimum of a set is defined analogously. Notice that S.2 and the associativity of $\sqcup$ imply that the space functions preserve the lub of any *finite* subset of *Con*. Recall (by Definition 2.1.14) that a space function that

preserves the supremum/infimum of any arbitrary subset of *Con* is said to be *join-complete/meet-complete*. Also recall that a space function in **C** is *continuous/downward continuous* if it preserves the supremum/infimum of any directed set/filtered set.

The join-completeness of space functions trivially implies their *(Scott) continuity*, a central concept in domain theory. From S.2 and the fact that constraint systems are complete lattices, the reverse implication is also true: Space continuity implies space completeness.

Proposition 3.3.1. *Let* $[\cdot]_i$ *be a space function of an scs. If* $[\cdot]_i$ *is continuous then* $[\cdot]_i$ *is join-complete.*

Proof. The above proposition follows from the fact that any function from a poset in which every non-empty finite supremum exists preserves arbitrary suprema if and only if it preserves both directed suprema and finite suprema [GHK+03]. □

3.4 The Extrusion Problem

Given an scs a legitimate question is whether it can be extended to an scse. In this section we would like to identify conditions that guarantee the existence of extrusion functions $\uparrow_1, \ldots, \uparrow_n$ for spaces $[\cdot]_1, \ldots, [\cdot]_n$ of any given n-scs.

From set theory we know that there is an extrusion function (i.e., a right inverse) $\uparrow_i$ for $[\cdot]_i$ iff $[\cdot]_i$ is *surjective*. Recall that the *fiber* of $y \in Y$, or *pre-image* of the singleton $\{y\}$, under $f : X \to Y$ is the set $f^{-1}(y) = \{x \in X \mid y = f(x)\}$. Thus the extrusion $\uparrow_i$ can be defined as a function, called *choice* function, that maps each element c to some element from the (non-empty because of surjectivity) fiber of c under $[\cdot]_i$. The existence of this choice function assumes, however, the Axiom of Choice.

Nevertheless, we are interested in an explicit construction for extrusion. This is possible for continuous space functions due to the following lemma stating that the fibers of space functions are directed sets. In fact, we can prove the lemma by showing something stronger: fibers are closed under finite joins.

Lemma 3.4.1 (Directed Fibers)**.** *Let* $\boldsymbol{C}$ *be an scs and let* $[\cdot]_i$ *be a surjective space function of* $\boldsymbol{C}$*. The fiber of any constraint* c *of* $\boldsymbol{C}$ *under* $[\cdot]_i$ *is a directed set.*

Proof. We prove that fibers are closed under finite joins. This trivially implies the lemma. Suppose a and b are in the fiber $[c]_i^{-1}$, that is $[a]_i = [b]_i = c$. We need to prove that $a \sqcup b \in [c]_i^{-1}$. Using S.2 we have $[a \sqcup b]_i = [a]_i \sqcup [b]_i = c \sqcup c = c$. Thus $a \sqcup b \in [c]_i^{-1}$. □

The following theorem, an immediate consequence of Lemma 3.4.1 and space continuity, identifies a sufficient condition to construct an extrusion function for the space $[\cdot]_i$ as the map that takes every c to the maximum of the fiber of c under $[\cdot]_i$.

Theorem 3.4.1 (Max Extrusion)**.** *Let* $\boldsymbol{C}$ *be an scs and let* $[\cdot]_i$ *be a surjective and continuous space function of* $\boldsymbol{C}$*. Then* $\uparrow_i : c \mapsto \bigsqcup [c]_i^{-1}$ *is a right inverse of* $[\cdot]_i$*.*

Proof. It follows from Lemma 3.4.1 that $[c]_i^{-1}$ is a directed set, thus because of continuity of the space function $[\uparrow_i c]_i = [\bigsqcup [c]_i^{-1}]_i = \bigsqcup \{[d]_i \mid d \in [c]_i^{-1}\} = \bigsqcup \{c\} = c$. □

It follows from the above theorem that any scs can be extended to scse if its space functions are continuous and surjective.

Local/Subjective Distribution

Notice that unlike space functions, extrusion functions are not required to preserve bottoms or binary lubs, i.e., they are not required to *distribute* over *finite* joins. In fact, the construction $\uparrow_i : c \mapsto \bigsqcup [c]_i^{-1}$ in Theorem 3.4.1 *may* result in $\uparrow_i \mathit{true} \neq \mathit{true}$ for some scs's. To better illustrate this situation, consider the following example.

Example 3.4.1. *Let* $\mathit{Con} = \mathbb{N} \cup \{\infty\}$ *and let* $\sqsubseteq$ *be the standard linear-order over* $\mathbb{N} \cup \{\infty\}$*. Let* $[\infty]_1 = \infty$ *and* $[n]_1 = \lfloor n/3 \rfloor$ *be a continuous and surjective space function. The tuple* $(\mathit{Con}, \sqsubseteq, [\cdot]_1)$ *is an scs with* $\mathit{true} = 0$*. We can apply Theorem 3.4.1 to obtain the extrusion function* $\uparrow_1 : c \mapsto \bigsqcup [c]_1^{-1}$ *for* $c \in \mathit{Con}$*. Notice, however* $\uparrow_1 0 = \bigsqcup [0]_1^{-1} = \bigsqcup \{0, 1, 2\} = 2 \neq 0$*.*

From a spatial point of view, however, any extrusion function $\uparrow_i$ distributes over finite joins if it is *within* a space $[\cdot]_i$; and from the epistemic point of view $\uparrow_i$ distributes over finite joins as far as agent i can tell. The following proposition states this formally

Let $[\cdot]_i$ be the space function of agent i in an scse. We write $c \approx_i d$ iff $[c]_i = [d]_i$. The equivalence relation $\approx_i$ is sometimes referred to as the *kernel* of $[\cdot]_i$. Intuitively, $c \approx_i d$ expresses the idea that c and d are equivalent to agent i.

Proposition 3.4.1. *Let* $\boldsymbol{C}$ *be an scse with constraints in Con, and let* $\uparrow_i$ *be the extrusion function of the agent* i *in* $\boldsymbol{C}$*. Then*

1. $\uparrow_i \mathit{true} \approx_i \mathit{true}$*, and*
2. $\uparrow_i(c \sqcup d) \approx_i \uparrow_i c \sqcup \uparrow_i d$ *for each* $c, d \in \mathit{Con}$*.*

Proof.

First we prove $\uparrow_i \mathit{true} \approx_i \mathit{true}$.

$$[\uparrow_i \mathit{true}]_i = \mathit{true} \tag{E.1}$$
$$[\uparrow_i \mathit{true}]_i = [\mathit{true}]_i \tag{S.1}$$
$$\uparrow_i \mathit{true} \approx_i \mathit{true}$$

We now prove $\uparrow_i(c \sqcup d) \approx_i \uparrow_i c \sqcup \uparrow_i d$.

$$[\uparrow_i(c \sqcup d)]_i = c \sqcup d \tag{E.1}$$
$$[\uparrow_i(c \sqcup d)]_i = [\uparrow_i c]_i \sqcup [\uparrow_i d]_i \tag{E.1}$$
$$[\uparrow_i(c \sqcup d)]_i = [\uparrow_i c \sqcup \uparrow_i d]_i \tag{S.2}$$
$$\uparrow_i(c \sqcup d) \approx_i \uparrow_i c \sqcup \uparrow_i d$$

□

Because the above distribution equalities depend on an agent, they can be regarded in spatial terms as being *local*, or in epistemic terms as being *subjective*. We consider next the *global/objective* version of the these equalities.

Global/Objective Distributed Extrusion

The condition E.2: $\uparrow_i true = true$ (i.e $\uparrow_i$ is strict) is not an unreasonable requirement since extruding or uttering *true* amounts to nothing regardless of the space context or the agent. In spatial terms E.2 should hold everywhere (*global*); in epistemic terms it should hold true regardless of the agent (*objective*). The same applies to the condition E.3: $\uparrow_i(c \sqcup d) = \uparrow_i c \sqcup \uparrow_i d$ (for every c and d) since it is not unreasonable to assume that extruding two pieces of information from the same space has the same effect as extruding them joined together. Notice that extrusion functions satisfying E.2 and E.3 distribute over finite joins; i.e., they preserve the supremum of finite sets. For this reason we shall refer to those extrusion functions satisfying E.2 and E.3 as being *(globally/objectively) distributed.*

Definition 3.4.1 (Spatial cs with Distributed Extrusion)**.** *A spatial constraint system with* distributed extrusion *(scs-de) is an scse* $(Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n, \uparrow_1, \ldots, \uparrow_n)$ *such that*

E.2 $\uparrow_i true = true$*, and*

E.3 $\uparrow_i(c \sqcup d) = \uparrow_i c \sqcup \uparrow_i d$ *for every* $c, d \in Con$*.*

We are also interested in the problem of extending scs's with distributed extrusion functions. For any continuous (and surjective) space function $[\cdot]_i$, the condition E.2 can be easily satisfied by a slight modification to the construction in Theorem 3.4.1: Take $\uparrow_i$ to be the function that maps c to *true* if $c = true$ else it maps c to $\bigsqcup [c]_i^{-1}$. The condition E.3, however, can be too strong of a requirement: There are surjective space functions for which *no inverse* satisfies E.3 –even if we assume the axiom of choice or restrict our attention to continuous space functions. Theorem 3.4.2 states this impossibility result.

Theorem 3.4.2 (Impossibility of Distributed Extrusion)**.** *There exists a surjective and continuous space function* $[\cdot]_i$ *of an scs* $(Con, \sqsubseteq)$ *such that: For every right inverse* g *of* $[\cdot]_i$ *there are* $c, d \in Con$ *such that* $g(c \sqcup d) \neq g(c) \sqcup g(d)$*.*

We describe the proof of Theorem 3.4.2 because it brings some insights into our next result. Consider the set $\mathbb{N} \cup \{\infty\}$ partially ordered as in the complete algebraic lattice in Figure 3.2. Let f be the self-map given by the arrows in Figure

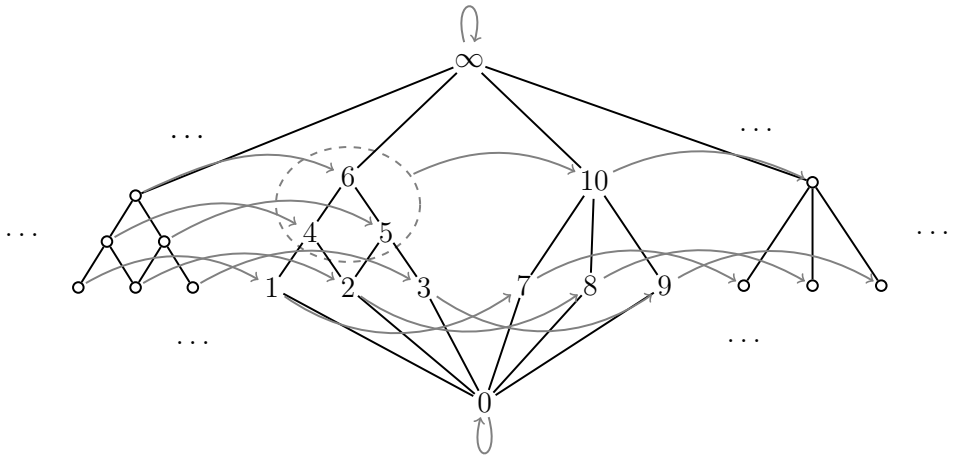


Figure 3.2 – A one-agent scs. Gray arrows depict a surjective and continuous space function over $\mathbb{N} \cup \{\infty\}$.

3.2. By examining this function, one can conclude that f is continuous and that it preserves finite joins (i.e., it satisfies S.1 and S.2). Hence the underlying lattice in Figure 3.2 is a one-agent spatial constraint system with f as space function. Notice that the fiber of 10 under f is $f^{-1}(10) = \{4, 5, 6\}$ and the fiber of any $e \in \mathbb{N} \cup \{\infty\}$ under f with $e \neq 10$ is a singleton set. This implies that there are exactly three different right inverse functions for f and they differ only on input 10. Name these functions g_4, g_5 and g_6 where $g_n(10) = n$. None of these functions satisfy E.3: We have $4 = g_4(10) = g_4(8 \sqcup 9) \neq g_4(8) \sqcup g_4(9) = 5$, then the symmetric case $5 = g_5(10) = g_5(7 \sqcup 8) \neq g_5(7) \sqcup g_5(8) = 4$, and finally $6 = g_6(10) = g_6(8 \sqcup 9) \neq g_6(8) \sqcup g_6(9) = 5$. This gives us a constructive witness f to the statement in Theorem 3.4.2.

Our strategy to prove Theorem 3.4.2 was to provide a space function with a fiber not closed under meets. In our particular construction the fiber of 10 under f is not closed under meets since $\sqcap\{4, 5, 6\} = 2$. We can prevent the existence of this kind of fibers by requiring space functions to be *meet-complete*. We conclude this section by showing that *meet-completeness* for space functions is in fact a *sufficient condition* for the existence of distributed extrusion functions.

Theorem 3.4.3 (Min Extrusion)**.** *Let $[\cdot]_i$ be any meet-complete and surjective space function of an scs. Then $\uparrow_i : c \mapsto \bigsqcap [c]_i^{-1}$ satisfies $[\uparrow_i c]_i = c$ (E.1), $\uparrow_i true =$*

true (E.2) and $\uparrow_i(c \sqcup d) = \uparrow_i c \sqcup \uparrow_i d$ *(E.3).*

Proof. Let us prove first $[\uparrow_i c]_i = c$. Since $[\cdot]_i$ is meet-complete we deduce

$$[\uparrow_i c]_i = [\bigsqcap [c]_i^{-1}]_i = \bigsqcap \{[d]_i \mid d \in [c]_i^{-1}\} = \bigsqcap \{c\} = c.$$

We now prove $\uparrow_i \mathit{true} = \mathit{true}$. From S.1 $\mathit{true} \in [\mathit{true}]_i^{-1}$ thus $\bigsqcap [\mathit{true}]_i^{-1} = \mathit{true}$.

Finally we need to prove $\uparrow_i(c \sqcup d) = \uparrow_i c \sqcup \uparrow_i d$. First we show $\uparrow_i$ is monotone.

Claim: Define $\uparrow_i : c \mapsto \bigsqcap [c]_i^{-1}$, if $c \sqsubseteq d$ then $\uparrow_i c \sqsubseteq \uparrow_i d$. The proof of this claim is by contradiction: Suppose that H.1 $c \sqsubseteq d$ but H.2 $\uparrow_i c \not\sqsubseteq \uparrow_i d$. Let us treat H.2 by cases. We use $c \parallel d$ to mean that c is not related to d, i.e., $(c,d),(d,c) \notin \sqsubseteq$.

— Assume that $\uparrow_i d \sqsubset \uparrow_i c$. We derive the following:

$$\begin{aligned}
[\uparrow_i d]_i &\sqsubseteq [\uparrow_i c]_i && \text{(Monotonicity of } [\cdot]_i\text{)}\\
d &\sqsubseteq c && \text{(E.1)}\\
d &= c && \text{(From H.1)}\\
\uparrow_i d &= \uparrow_i c && \text{(A contradiction with } \uparrow_i d \sqsubset \uparrow_i c.\text{)}
\end{aligned}$$

— Assume $\uparrow_i c \parallel \uparrow_i d$. Because $[\cdot]_i$ is meet complete we have $[\uparrow_i c \sqcap \uparrow_i d]_i = [\uparrow_i c]_i \sqcap [\uparrow_i d]_i$. Applying E.1 and using hypothesis H.1 we obtain $[\uparrow_i c]_i \sqcap [\uparrow_i d]_i = c \sqcap d = c$, therefore

$$\uparrow_i c \sqcap \uparrow_i d \in [c]_i^{-1}. \tag{3.4.1}$$

From Equation 3.4.1 and the definition of $\uparrow_i$ we conclude $\uparrow_i c \sqsubseteq \uparrow_i c \sqcap \uparrow_i d$. But this contradicts $\uparrow_i c \sqcap \uparrow_i d \sqsubset \uparrow_i c$ which follows from the hypothesis $\uparrow_i c \parallel \uparrow_i d$.

This concludes the proof of the claim. We can now prove $\uparrow_i(c \sqcup d) = \uparrow_i c \sqcup \uparrow_i d$.

Since $c, d \sqsubseteq c \sqcup d$, from the monotonicity of $\uparrow_i$ we have $\uparrow_i c, \uparrow_i d \sqsubseteq \uparrow_i(c \sqcup d)$, therefore $\uparrow_i c \sqcup \uparrow_i d \sqsubseteq \uparrow_i(c \sqcup d)$. Furthermore applying S.2 and E.1 we obtain $[\uparrow_i c \sqcup \uparrow_i d]_i = c \sqcup d$, thus $\uparrow_i c \sqcup \uparrow_i d \in [c \sqcup d]_i^{-1}$. Since $\uparrow_i(c \sqcup d) = \bigsqcap [c \sqcup d]_i^{-1}$ we derive $\uparrow_i(c \sqcup d) \sqsubseteq \uparrow_i c \sqcup \uparrow_i d$ which concludes the proof. □

Therefore any spatial cs whose space functions are meet-complete and surjective can be extended to an scse with distributed extrusion by defining $\uparrow_i c$ as the map $c \mapsto \bigsqcap [c]_i^{-1}$.

Remark 3.4.1. *Notice that from Proposition 3.3.1, Theorem 3.4.1 and Theorem 3.4.3 are dual in the sense that whereas Theorem 3.4.1 requires space functions to be* join-complete*, Theorem 3.4.3 requires space functions to be* meet-complete*.*

3.5 Properties of Space and Extrusion

In what follows we discuss some distinctive properties of space and extrusion. An immediate consequence of the definition of scse's is that their spatial and extrusion functions must be surjective and injective, respectively.

Corollary 3.5.1. *Let $[\cdot]_i$ and $\uparrow_i$ be space and extrusion functions of an scse. Then $[\cdot]_i$ is surjective and $\uparrow_i$ is injective.*

Proof. Axiom E.1 implies that there exists $y = \uparrow_i c$ for every $c \in Con$ such that $[y]_i = c$. This proves surjectivity. Now, as a means of contradiction assume that $\uparrow_i$ is not injective. Then there exists elements $c \neq d$ such that $\uparrow_i c = \uparrow_i d$. But then $[\uparrow_i c]_i = [\uparrow_i d]_i$. Applying E.1 we obtain $c = d$, a contradiction. □

Remark 3.5.1. *If $[\cdot]_i$ is surjective, then for all $a \in Con$ there exists $c \in Con$ where $a = [c]_i$ for any $i \in \{1, \ldots, n\}$ in a n-scse. Now, this means there exist $c \in Con$ and $d \in Con$ s.t. $[c]_i = a = [d]_j$ for all $1 \leq i, j \leq n$. In an epistemic context, this property might seem unusual at first look and even hint to an equality of seemingly unrelated beliefs from different agents. However, beliefs c and d are indeed related and $d = \uparrow_j [c]_i$ is their actual relation.*

Consistent and Contradicting Agents

The following property of spatial constraint systems with extrusion has a noteworthy epistemic interpretation. Notice that in scs's nothing prevented us from having $[\mathit{false}]_i \neq \mathit{false}$. Intuitively, inconsistencies generated by an agent may be confined within its own space. In scs's with extrusion, however, the agents' ability to move information outside their spaces prevents inconsistency confinement. This has a pleasant correspondence with epistemic logic since $[\mathit{false}]_i = \mathit{false}$ reflects the principle, referred to as the *Consistency Axiom* in *belief/doxastic logics*, that no agent can possibly believe the false statement.

Property 3.5.1 (Space Consistency)**.** *Let* $[\cdot]_i$ *be a space function of an scse. Then* $[\mathit{false}]_i = \mathit{false}$*.*

Proof. We derive the following

$$\begin{aligned}
[\mathit{false}]_i &= [\mathit{false} \sqcup \uparrow_i \mathit{false}]_i && (\mathit{false} \sqcup \cdot = \mathit{false}) \\
&= [\mathit{false}]_i \sqcup [\uparrow_i \mathit{false}]_i && \text{(S.2)} \\
&= [\mathit{false}]_i \sqcup \mathit{false} && \text{(E.1)} \\
&= \mathit{false} && (\cdot \sqcup \mathit{false} = \mathit{false})
\end{aligned}$$

□

Nevertheless, for $i \neq j$ we allow the following to occur in an scse: $[c]_i \sqcup [d]_j \neq \mathit{false}$, even when $c \sqcup d = \mathit{false}$. Thus we may have agents whose information is inconsistent with that of others. This reflects the distributive and epistemic nature of the agents as they may have different information about the same incident or have *contradicting beliefs*.

Orders

The next properties involve notions from order theory [DP02]. They will allows us to infer properties of information from placing into a space or extruding it. For example, to infer $c \sqsubseteq d$ from observing $f(c) \sqsubseteq f(d)$ where f is either a space or a extrusion function. Recall Definition 2.1.3 and consider the next definition:

Definition 3.5.1. *Given* $(\mathit{Con}, \sqsubseteq)$ *a self-map* f *over* Con *is said to be* order-automorphism *if it is a surjective order-embedding. Additionally, we say that* f *is* strictly monotonic *(or strict-order preserving) if* $c \sqsubset d$ *implies* $f(c) \sqsubset f(d)$*.*

From E.3 it follows that globally distributed extrusion functions preserve $\sqsubseteq$ (monotonicity). From the axioms S.2 and E.1 one can also show that they reflect $\sqsubseteq$. Thus, extrusion functions are order-embeddings:

Property 3.5.2 (Extrusion Embedding)**.** *Let* $\uparrow_i$ *be a distributed extrusion function of an scse. Then* $\uparrow_i$ *is an order-embedding.*

Proof. Since $\uparrow_i$ preserves binary joins (E.3) it follows that $\uparrow_i$ is order-preserving. To prove that $\uparrow_i$ is order-reflecting suppose $\uparrow_i c \sqsubseteq \uparrow_i d$. Then, by monotonicity of the space functions (Remark 2.2.2) we obtain $[\uparrow_i c]_i \sqsubseteq [\uparrow_i d]_i$ and using E.1 we conclude $c \sqsubseteq d$. □

Analogous to inconsistency confinement, we could have $[c]_i = [d]_i$ for some c and d such that $c \neq d$. As we already mentioned this could be interpreted as saying that agent i cannot distinguish c from d; i.e., $c \approx_i d$. For some meaningful constraint systems, space functions necessarily preserve distinctness, i.e., they are *injective*. In particular,

Proposition 3.5.1 (Injective Spaces)**.** *Let $[\cdot]_i$ be a space function of an scse $(Con, \sqsubseteq)$. Then $[\cdot]_i$ is injective if (1) Con is a finite set, or if (2) $[\cdot]_i$ is strictly monotonic.*

Proof. Suppose *Con* is a finite set. Then the injectivity of $[\cdot]_i$ follows from the pigeon hole principle given surjectivity of the self-map $[\cdot]$ and the fact that *Con* is finite. Now suppose $[\cdot]_i$ is strictly monotonic. Let c, d with $c \neq d$. We need to prove that $[c]_i \neq [d]_i$. We have two cases:

— $c \sqsubset d$. Since $[\cdot]_i$ is strictly monotonic then $[c]_i \sqsubset [d]_i$ therefore $[c]_i \neq [d]_i$.

— $c \parallel d$ (i.e., $(c, d), (d, c) \notin \sqsubseteq$). Then we have $c \sqsubset c \sqcup d$, thus by using strict monotonicity and S.2 we obtain $[c]_i \sqsubset [c]_i \sqcup [d]_i$. We conclude $[c]_i \neq [d]_i$.

□

Like extrusion functions, injective space functions of scse also preserve and reflect the order. Furthermore since they are surjective, we conclude the following.

Property 3.5.3 (Automorphic Spaces)**.** *Let $[\cdot]_i$ be an injective space function of an scse. Then $[\cdot]_i$ is an order automorphism.*

Proof. An automorphism is defined as a bijective self-map that is also an order-embedding. As $[\cdot]_i$ is a surjective function (Corollary 3.5.1), and by hypothesis it is injective, then it is also a bijection. Any space function is order-preserving since they preserve binary joins (Remark 2.2.2). It remains to prove that $[\cdot]_i$ is order-reflecting.

Suppose $[c]_i \sqsubseteq [d]_i$. Using S.2 and the hypothesis we obtain $[c]_i \sqcup [d]_i = [c \sqcup d]_i = [d]_i$. From the injectivity of $[\cdot]_i$, $c \sqcup d = d$, thus $c \sqsubseteq d$. □

A noteworthy corollary of Property 3.5.3 is that injective space functions are *Scott-continuous* (in fact meet and joint-complete) since order automorphisms are known to preserve whatever infima and suprema may exist in the corresponding poset [Goo10].

Corollary 3.5.2 (Complete Spaces)**.** *Let $[\cdot]_i$ be a space function of an scse $(Con, \sqsubseteq)$. If $[\cdot]_i$ is an automorphism then $[\cdot]_i$ is join-complete and also meet-complete.*

Notice that from Proposition 3.5.1, Corollary 3.5.2, and Property 3.5.3 we conclude that any *strictly monotonic* space function of an scse is continuous. Any space function of an scse is surjective and it has a property that is *stronger* than monotonicity: Namely it preserves finite joins (Remark 2.2.2). One may then wonder if space functions from scse's are already continuous. A negative answer is given in the example below.

Example 3.5.1 (Lexical Order)**.** *Let $Con = \mathbb{N} \times \mathbb{N} \cup \{(\infty, \infty)\}$ and let $\sqsubseteq$ be the obvious lexical order on Con. Notice $(Con, \sqsubseteq)$ is a complete algebraic lattice. The function $[\cdot]_1$ is given by $[(\infty, \infty)]_1 = (\infty, \infty)$, $[(0, n)]_1 = (0, 0)$, $[(1, n)]_1 = (0, n+1)$ and $[(m, n)]_1 = (m-1, n)$ for every $n, m \in \mathbb{N}$ with $m \geq 2$. Clearly $[\cdot]_1$ satisfies S.1 and S.2. Furthermore $[\cdot]_1$ is meet-complete and surjective, so Theorem 3.4.3 gives us a distributed extrusion function $\uparrow_1 : (n, m) \mapsto \bigsqcap [(n, m)]_i^{-1}$. Therefore $(Con, \sqsubseteq, [\cdot]_1, \uparrow_1)$ is an scs with distributed extrusion. Nevertheless $[\cdot]_1$ is not continuous: Take the directed set $S = \{(0, n) \mid n \geq 0\}$. We have $[\bigsqcup S]_1 = [(1, 0)]_1 = (0, 1) \neq (0, 0) = \bigsqcup \{[(0, n)]_1 | \ \ n \geq 0\}$.* □

The above example also shows an application of Theorem 3.4.3 to derive an extrusion function for a rather simple scs. Notice that we could not have applied Theorem 3.4.1 because the $[\cdot]_1$ was shown not to be continuous. In the Application section we will derive extrusion functions for a meaningful and more involved scs using Theorem 3.4.1.

3.6 Galois Connections

We conclude this section by stating a pleasant correspondence between space and extrusion. In Example 3.5.1 we used Theorem 3.4.3 to derive extrusion. This theorem tells us that we can extend any spatial cs whose space functions are meet-complete and surjective to an scse with distributed extrusion by defining $\uparrow_i c$ as the map $c \mapsto \bigsqcap [c]_i^{-1}$. From order theory we know that with such a definition we obtain a *(monotone) Galois connection* between space and extrusion.

Let us remember Definition 2.1.4. Given $(Con, \sqsubseteq)$, we say that a pair (l, u) of monotone self-maps on Con is a *Galois connection* iff $l(c) \sqsubseteq d \Leftrightarrow c \sqsubseteq u(d)$ for every $c, d \in Con$. In a Galois connection (l, u), l and u are called the *lower* and *upper adjoint*, respectively. The following property follows directly from the theory of adjoints [AJ94].

Property 3.6.1 (Galois Connections)**.** *Let $[\cdot]_i$ and $\uparrow_i$ be the space and extrusion function for agent i in an scs with distributed extrusion $(Con, \sqsubseteq)$. Then $(\uparrow_i, [\cdot]_i)$ is a Galois connection if and only if $\uparrow_i c = \bigsqcap [c]_i^{-1}$ for every $c \in Con$. Similarly, $([\cdot]_i, \uparrow_i)$ is a Galois connection if and only if $\uparrow_i c = \bigsqcup [c]_i^{-1}$ for every $c \in Con$.*

Proof. Here we appeal to the theory of adjoints. We adapt proposition 3.1.10 from [AJ94] to self-maps on Con: Given two monotonic self-maps $l : Con \to Con$ and $u : Con \to Con$ the following are equivalent:

1. $\forall c \in Con : l(c) = \bigsqcap u^{-1}(c)$
2. $\forall c \in Con : u(c) = \bigsqcup l^{-1}(c)$
3. $\forall c, d \in Con : c \sqsubseteq u(d)$ iff $l(c) \sqsubseteq d$ (i.e., (l, u) is a Galois connection).

The functions $[\cdot]_i$ and $\uparrow_i \cdot$ are monotonic because they preserve binary joins (S.2, E.3). By taking $l = \bigsqcap u^{-1}(c) = \uparrow_i = \bigsqcap [c]_i^{-1}$ we obtain (1). By taking $u = \bigsqcup l^{-1}(c) = \uparrow_i = \bigsqcup [c]_i^{-1}$ we obtain (2). □

It follows from Property 3.6.1 that the pair $(\uparrow_1, [\cdot]_1)$ in Example 3.5.1 is a Galois connection. The following is a simple example of a space and extrusion pair that can be shown *not to be* a Galois connection using Property 3.6.1.

Example 3.6.1. *Let $Con = \mathbb{N} \cup \{\infty\}$ and let $\sqsubseteq$ be the standard linear-order over $\mathbb{N} \cup \{\infty\}$. Let $[\infty]_1 = \infty = \uparrow_1 \infty$, $[0]_1 = 0 = \uparrow_1 0$ and $[n]_1 = \lceil n/3 \rceil$ and*

$\uparrow_1 n = 3n - 1$ *for any* $n \in \mathbb{N} - \{0\}$. *The tuple* $(Con, \sqsubseteq, [\cdot]_1, \uparrow_1)$ *is an scs with distributed extrusion. But* $2 = \uparrow_i 1 \neq \bigsqcap [1]_i^{-1} = 1$, *hence from Property 3.6.1 we can conclude that* $(\uparrow_i, [\cdot]_i)$ *is not a Galois connection. (One can also verify using Property 3.6.1 that the reversed pair* $([\cdot]_i, \uparrow_i)$ *is not a Galois connection either.)* $\square$

Recall that $e \sqsubseteq e'$ can be thought of as the entailment of e by e'. A Galois connection of the form

$$[c]_i \sqsubseteq d \Leftrightarrow c \sqsubseteq \uparrow_i d \tag{3.6.1}$$

reduces entailment of space to the entailment by extrusion. We will see an application of this observation in the next section.

3.7 Summary

We considered the problem of deriving extrusion for space functions and we identified sufficient conditions of space functions to obtain explicit constructions for extrusion. Surjectivity of space functions is of course a necessary condition for the existence of the corresponding extrusion functions. If the space function is join-complete (or continuous), Theorem 3.4.1 gives us a construction that maps each constraint to the least upper bound of its pre-image (or fiber) under the space function. If the space function is meet-complete Theorem 3.4.3 gives us a dual construction that maps each constraint to the greatest lower bound of its pre-image under the space function.

In what follows from this thesis we use some of the above results to derive extrusion for some modal operators. For modal logics in general, the underlying constraint system we use is Kripke's (defined later in Definition 4.2.1) and thus space functions correspond to the box ($\Box$) operator, the join correspond to conjunction, and meet correspond to disjunction (see Definition 4.2.2). If it exists, the extrusion operator would be a reverse modality, say $\Box^{-1}$, such that $\Box\Box^{-1}\phi$ is logically equivalent to ϕ. Theorem 3.4.1 always applies since the space functions of any Kripke constraint system are join-complete (same argument from Remark 2.2.1). In fact the box operator always distributes over conjunction but not always over disjunction.

Nevertheless, there exist cases where space functions are also meet-complete. For example the box operator being interpreted as the *next modality* $\bigcirc$ of temporal logic [PM92] since it distributes over both conjunction and disjunction. In this case we could apply both Theorem 3.4.1 and Theorem 3.4.3. Interestingly, we would obtain two different but well-known reverse modalities for the next operator. Theorem 3.4.1 would give us a *strong-previous* modality $\ominus$ while Theorem 3.4.3 would give us a *weak-previous* modality $\widetilde{\ominus}$. Notice that unlike Theorem 3.4.1, Theorem 3.4.3 guarantees that the derived extrusion preserves the bottom element *true*. In fact, the temporal formula $\widetilde{\ominus}\mathrm{T}$ is logically equivalent to T while $\ominus\mathrm{T}$ is not (since $\ominus p$ is false at time 0 for any p). This interpretation of space as a time operator will be expanded thoroughly in Chapter 6.

Finally, we presented distinctive properties of space and extrusion. Property 3.5.1 tells us a fundamental aspect of extrusion; unlike general spatial constraint systems, the space functions of scs with extrusion cannot confine inconsistencies. For example for Kripke constraint systems, where space functions correspond to the box ($\Box$) operator if they admit extrusion then $\Box\mathrm{F}$ must be equivalent to F. Proposition 3.5.1 identifies conditions under which spaces of scs with extrusion must preserve distinctiveness of information. The other properties state the preservation and reflection of the underlying order/entailment relation w.r.t space and extrusion. These properties allow us to infer entailment between some given information from the entailment when the information is placed in some agent's space or when it is extruded. Finally, the Galois connections between space and extrusion allow us to reduce the entailment of/by spatial information from entailment by/of extruded information. All the results presented in this chapter were published in [HPRV15].

Chapter 4

Opinions & Lies: A Bimodal Logic

4.1 Modal Logics

In this section we present the basic concepts we use in this chapter from modal logics. See [BDRV02] for a more thorough introduction to modal logic and their applications. We begin by defining what a modal language is.

Definition 4.1.1 (Modal language)**.** *Let Φ be a set of primitive propositions. We denote by $\mathcal{L}_n(\Phi)$ the modal language made of formulas using the following syntax:*

$$\phi, \psi, \ldots := p \mid \phi \wedge \psi \mid \neg\phi \mid \Box_i \phi$$

where $p \in \Phi$ and $i \in \{1, \ldots, n\}$.

Notation 4.1.1. *Disjunction and implication are defined in the standard way; $\phi \vee \psi \stackrel{\texttt{def}}{=} \neg(\neg\phi \wedge \neg\psi)$ and $\phi \Rightarrow \psi \stackrel{\texttt{def}}{=} \neg\phi \vee \psi$. We also define a dual of the modality; $\Diamond_i \phi \stackrel{\texttt{def}}{=} \neg\Box_i\neg\phi$.*

Having defined a modal language, we can also define a semantic model to give *meaning* to the language. In other words, this enables us to represent a formula from the language in some abstract model.

Definition 4.1.2 (Kripke Structures)**.** *An n-agent Kripke structure (KS) M over a set of atomic propositions Φ is a tuple:*

$$M = (S, \pi, \mathcal{R}_1, \ldots, \mathcal{R}_n) \tag{4.1.1}$$

where:

— *S is a nonempty set of states,*

— *$\pi : S \to (\Phi \to \{0,1\})$ is an interpretation that associates with each state a truth assignment to the primitive propositions in Φ, and*

— *$\mathcal{R}_i$ is a binary relation on S.*

Notation 4.1.2. *The states of a KS are often referred to as* worlds. *Each $\mathcal{R}_i$ is referred to as the* accessibility *or* possibility *relation for agent i: $(s,t) \in \mathcal{R}_i$ is meant to capture that agent i considers world t possible given its information in world s. We use $s \xrightarrow{i}_M t$ to denote $(s,t) \in \mathcal{R}_i$ in the KS M. We use $\mathcal{W}_i(M,s) = \{t \mid s \xrightarrow{i}_M t\}$ to denote the worlds agent i considers possible from a state s of KS M. Alternatively, we write $\mathcal{W}_i^{-1}(M,s) = \{t \mid t \xrightarrow{i}_M s\}$ to denote the worlds from which agent i considers state s possible. The interpretation function π tells us what primitive propositions are true at a given world: p holds at state s iff $\pi(s)(p) = 1$. We use π_M to denote the interpretation π of the KS M.*

We can define a notion of semantic validity using a relation between the models and the formulas. We say that ϕ *holds* at (M,s) whenever $(M,s) \models \phi$. Furthermore, ϕ *is valid* in M if ϕ holds in all states of M and ϕ *is satisfied* in M if there exists a state in M where ϕ holds.

Definition 4.1.3 (Kripke semantics)**.** *We define the relation $\models$ inductively as follows:*

- $(M,s) \models p$ *iff* $\pi_M(s)(p) = 1$
- $(M,s) \models \phi \wedge \psi$ *iff* $(M,s) \models \phi$ *and* $(M,s) \models \psi$
- $(M,s) \models \neg\phi$ *iff* $(M,s) \not\models \phi$
- $(M,s) \models \Box_i\phi$ *iff* $(M,t) \models \phi$ *for all* $t \in \mathcal{W}_i(M,s)$

Theorem 4.1.1. *The semantic validity of disjunction, implication and the dual modality are as follows:*

- $(M,s) \models \phi \vee \psi$ *iff* $(M,s) \models \phi$ *or* $(M,s) \models \psi$
- $(M,s) \models \phi \Rightarrow \psi$ *iff* *whenever* $(M,s) \models \phi, (M,s) \models \psi$ *holds*
- $(M,s) \models \Diamond_i\phi$ *iff* *there exists* $t \in \mathcal{W}_i(M,s)$ *such that* $(M,s) \models \phi$

Consider a class of Kripke structures $\mathbb{M}$. We can expand the definition of validity and satisfiability accordingly; ϕ *is valid* in $\mathbb{M}$ if ϕ is valid in all KS of $\mathbb{M}$ and ϕ *is satisfied* in $\mathbb{M}$ if there exists a KS in $\mathbb{M}$ that satisfies ϕ.

In order to formally prove a statement, a logic can have an associated *axiom system* AX. This is a set of *inference rules* of the form "from ϕ infer ψ" and a set of *axioms* which are just formulas. A proof of a formula ϕ in AX is a sequence of formulas ending with the formula ϕ and each step is either an instance of an axiom or an application of an inference rule. We write $AX \vdash \phi$ if there exists such sequence for ϕ to say that ϕ is *provable/true* in AX.

Example 4.1.1 (Axiom System $\mathbf{K}_n$)**.** *The axiom system* $\mathbf{K_n}$ *is made up of two axioms and two inference rules:*

A.1 All tautologies from propositional logic,

A.2 $\Box_i(\phi \Rightarrow \psi) \Rightarrow (\Box_i\phi \Rightarrow \Box_i\psi)$ *for all* $i = 1 \dots n$,

R.1 From ϕ *and* $\phi \Rightarrow \psi$ *infer* ψ,

R.2 From ϕ *infer* $\Box_i\phi$ *for all* $i = 1 \dots n$.

A.2 is usually called the *distribution axiom*, rule R.1 is *modus ponens* and R.2 is called the *generalization* or *necessitation* rule. A set of modal formulas Λ is called a *normal modal logic* if it contains the axioms system $\mathbf{K}_n$ (or $\mathbf{K}$ when n is unimportant).

Finally, we formally state the link between the axiomatic and the semantic world. This correspondence is important in the sense that it ties what is *valid* (semantics) to what is *true* (axioms) in a logic.

Definition 4.1.4 (Soundness and completeness)**.** *A logic* Λ *with axiom system* AX *is said to be* sound *wrt to semantic class* $\mathbb{M}$ *iff for every* ϕ *we have that* $AX \vdash \phi$ *implies* $\mathbb{M} \models \phi$*. Alternatively, a logic* Λ *with axiom system* AX *is* complete *wrt to semantic class* $\mathbb{M}$ *iff for every* ϕ *we have that* $\mathbb{M} \models \phi$ *implies* $AX \vdash \phi$*.*

4.2 Kripke Spatial Constraint Systems

We now revisit a concrete spatial constraint system from [KPPV12]. This constraint system will play a significant role in the rest of the document. We basically

extend Example 2.2.2 by moving from Boolean assignments to KS. Other examples of spatial constraint system for epistemic reasoning are Aumann structures [KPPV12].

Recall that in Example 2.2.2 constraints are sets of boolean assignments. This allowed us to interpret each propositional formula as a constraint; the set of assignments that are models of (or satisfy) the formula. Similarly, in the following example (spatial) constraints are sets of (pointed) KS models. A *pointed KS* is a pair (M, s) where M is a KS and s, called the *actual world*, is a state of M. This will allows us to interpret each modal formula as its set of pointed KS models; i.e., a spatial constraint.

Definition 4.2.1 (Kripke scs [KPPV12])**.** *Let $\mathcal{S}_n(\Phi)$ be a non-empty set of n-agent Kripke structures over Φ. Let Δ be the set of all pointed Kripke structures (M, s) such that $M \in \mathcal{S}_n(\Phi)$. We define the Kripke n-scs for $\mathcal{S}_n(\Phi)$ as*

$$\boldsymbol{K}(\mathcal{S}_n(\Phi)) = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$$

where $Con = \mathcal{P}(\Delta)$, and for every $X, Y \in Con : X \sqsubseteq Y$ iff $Y \subseteq X$, and

$$[X]_i = i(X) \text{ where } i(X) \stackrel{\texttt{def}}{=} \{(M, s) \in \Delta \mid \forall t : s \xrightarrow{i}_M t \text{ implies } (M, t) \in X\} \tag{4.2.1}$$

for every agent $i \in \{1, \ldots, n\}$.

The scs $\mathbf{K}(\mathcal{S}_n(\Phi))$ is a complete algebraic lattice given by a powerset ordered by $\supseteq$. The $\sqcup$ is set intersection, the top element *false* is $\emptyset$, and bottom *true* is the set Δ of all pointed Kripke structures (M, s) with $M \in \mathcal{S}_n(\Phi)$. Similar to Example 2.2.2, a constraint c in $\mathbf{K}(\mathcal{S}_n(\Phi))$ is compact iff $\Delta \setminus c$ is a finite set [KPPV12].

Proposition 4.2.1. *Let $\boldsymbol{K}(\mathcal{S}_n(\Phi)) = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$ be a Kripke n-scs. Each $[\cdot]_i$ is a space function.*

Proof. We show that $[\cdot]_i$ fulfills the axioms S.1 and S.2:

For S.1:

$[true]_i$ by definition is $[\Delta]_i = \{(M, s) \in \Delta \mid \forall t : s \xrightarrow{i}_M t \text{ implies } (M, t) \in \Delta\}$. Nonetheless every $(M, s) \in \Delta$ consequently $[\Delta]_i = \Delta$ and $[true]_i = true$.

For S.2:

$$\begin{aligned}
[c \sqcup d]_i = & \quad \left\{(M,s) \in \Delta \mid \forall t : s \xrightarrow{i}_M t \text{ implies } (M,t) \in c \sqcup d\right\} \\
= & \quad \left\{(M,s) \in \Delta \mid \forall t : s \xrightarrow{i}_M t \text{ implies } (M,t) \in c \cap d\right\} \\
= & \quad \left\{(M,s) \in \Delta \mid \forall t : s \xrightarrow{i}_M t \text{ implies } (M,t) \in c\right\} \cap \\
& \quad \left\{(M,s) \in \Delta \mid \forall t : s \xrightarrow{i}_M t \text{ implies } (M,t) \in d\right\} \\
= & \quad [c]_i \cap [d]_i \\
= & \quad [c]_i \sqcup [d]_i
\end{aligned}$$

□

Similarly to Remark 3.2.1, since in Kripke scs meets are unions and joins intersections then Kripke scs are also frames. Also, because Kripke scs are of the form $(\mathcal{P}(U), \supseteq)$ (see Example 2.1.1), the operators $\rightarrow$ and $\sim$ coincide with the constructions $\Rightarrow$ and $\neg$ defined in Remark 2.2.1.

A modal language

Modal formulae can be interpreted as constraints in the scs $\mathbf{K}(\mathcal{S}_n(\Phi))$. This kind of interpretation will be frequently used in the rest of this thesis. Recall that the modal language $\mathcal{L}_n(\Phi)$ in Equation 4.1.1 is obtained by extending the grammar for the propositional language $\mathcal{L}_0(\Phi)$ in Equation 2.2.1 with modalities $\Box_i\phi$ in the standard way.

Definition 4.1.3 gives us the semantics of modal logics using KS's. Therefore, as in Example 2.2.2 we can interpret each formula ϕ as constraints in Kripke constraint systems.

Definition 4.2.2 (Kripke Constraint Interpretation)**.** *Let $\boldsymbol{K}(\mathcal{S}_n(\Phi))$ be a Kripke scs denoted by $\mathcal{C}$. Given a modal formula ϕ in the language $\mathcal{L}_n(\Phi)$, its interpreta-*

tion in the Kripke scs $\mathcal{C}$ is the constraint $\mathcal{C}[\![\phi]\!]$ inductively defined as follows:

$$\begin{aligned}
\mathcal{C}[\![p]\!] &= \{(M,s) \in \Delta |\ \pi_M(s)(p) = 1\ \} \\
\mathcal{C}[\![\phi \wedge \psi]\!] &= \mathcal{C}[\![\phi]\!] \sqcup \mathcal{C}[\![\psi]\!] \\
\mathcal{C}[\![\neg\phi]\!] &= \Delta \setminus \mathcal{C}[\![\phi]\!] \\
\mathcal{C}[\![\Box_i\phi]\!] &= [\ \mathcal{C}[\![\phi]\!]\ ]_i
\end{aligned}$$

where Δ is the set of all pointed Kripke structures (M,s) such that $M \in \mathcal{S}_n(\Phi)$.

Notation 4.2.1. *Notice that the interpretation of $\Box_i(\phi)$, $\mathcal{C}[\![\Box_i(\phi)]\!]$, is equal to the constraint $[\,\mathcal{C}[\![\phi]\!]\,]_i$ in $\boldsymbol{K}(\mathcal{S}_n(\Phi))$. Often, by abuse of notation, we shall suppress the semantic symbols $\mathcal{C}[\![\ ]\!]$ from formulae–e.g., we write $[\phi]_i$ for the constraint $[\,\mathcal{C}[\![\phi]\!]\,]_i$.*

Following our intended meaning of constraints, we think of $[\phi]_i$ as stating that ϕ holds in the space of agent i, or as an epistemic assertion stating that agent i considers/believes ϕ to be true.

4.3 A Logic of Belief and Utterance

In Section 3.4 we discussed the problem of constructing extrusion functions for spatial constraint systems. In this section we want to derive explicit extrusion functions for a meaningful family of the Kripke scs (Definition 4.2.1) as an application of the results we obtained in Sections 3.4 and 3.5.

Recall that we can associate a modal language (Definition 4.1.1) to a Kripke scs by interpreting formulae as constraints, i.e., set of pointed Kripke structures. Under such association, $\Box_i\phi = [\phi]_i$ states that ϕ holds true in the space of i (Notation 4.2.1). Finding an extrusion $\uparrow_i$ for each $[\cdot]_i$ will also allow us to derive an *inverse* modality for $\Box_i$. We will use the derived modality to specify utterances and lies with a modal language.

Let us also recall the (n-agent) Kripke scs $\mathbf{K}(\mathcal{S}_n(\Phi)) = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$ in Definition 4.2.1. This scs is parametric in a set of (n-agents) Kripe structures $\mathcal{S}_n(\Phi)$ defined over a set of primitive propositions Φ. Its set of constraints is defined as $Con = \mathcal{P}(\Delta)$ where Δ is the set of all pointed KS (M,s) such that $M \in \mathcal{S}_n(\Phi)$, $\sqsubseteq$ is reversed set inclusion, the join operation $\sqcup$ is set intersection,

the meet operation $\sqcap$ is set union, the top element *false* is $\emptyset$, and its bottom *true* is Δ. The space functions are given by :

$$[c]_i \stackrel{\texttt{def}}{=} \{(M,s) \in \Delta \mid \forall t : \text{ if } s \xrightarrow{i}_M t \text{ then } (M,t) \in c\} \tag{4.3.1}$$

for each $i \in \{1, \ldots, n\}$.

We will thus endeavor to find suitable Kripke structures for scs such that extrusions for each space function can be constructed. The path we follow for this is: (1) we restrict KS to left-total accessibility and show this ensures space consistency, (2) KS are then further restricted to left-unique accessibility relations to guarantee surjectivity of space functions, (3) we show these to be also continuous and, finally (4) we apply theorem 3.4.1 to derive an extrusion for each space function over the restricted KS's.

Left-total left-unique Kripke Structures

Modal logics are typically interpreted over different classes of KS that are usually obtained by imposing conditions on their accessibility relations (see Section 5.1 for examples of this). We denote with $\mathcal{M}$ the class of KS whose accessibility relations are unrestricted, thus formulae should be interpreted as elements of the Kripke scs $\mathbf{K}(\mathcal{M}_n(\Phi))$ where $\mathcal{M}_n(\Phi)$ is the set of *all* n-agents KS's over Φ.

Notation 4.3.1. *For notational convenience, we take the set Φ of primitive propositions and n to be fixed from now on and omit them from the notation. E.g., we write $\mathcal{M}$ instead of $\mathcal{M}_n(\Phi)$.*

We say that a set $\mathcal{S}$ of KS's *satisfies space consistency* iff $[\mathit{false}]_i = \mathit{false}$ for every space function $[\cdot]_i$ in $\mathbf{K}(\mathcal{S})$. It follows from Property 3.5.1 that space consistency is a *necessary condition* for the existence of extrusion functions.

Let us begin with $\mathbf{K}(\mathcal{M})$. We can verify that this scs does not satisfy space consistency. First recall from Notation 4.1.2 that $\mathcal{W}_i(M,s) = \{t \mid s \xrightarrow{i}_M t\}$ denote the worlds agent i considers possible from the world s of KS M. Take a pointed KS (M', s') such that $\mathcal{W}_i(M', s') = \emptyset$. Notice that in $\mathbf{K}(\mathcal{M})$, $\mathit{false} = \emptyset$. From Equation 4.3.1 we conclude that $(M', s') \in [\mathit{false}]_i$ thus violating space consistency. Property 3.5.1 then tells us that $\mathbf{K}(\mathcal{M})$ cannot be extended to an scs with extrusion.

Left-total KS's Let us consider more restricted sets of KS's. We already mentioned, in the preamble of Property 3.5.1, the connection between space consistency and the Consistency Axiom. The condition on KS associated with the Consistency Axiom is that of being *left-total*. An accessibility relation $\mathcal{R}_i$ of agent i in a KS M is said to be *left-total* (or *serial*) if for every s there exists t such that $(s,t) \in \mathcal{R}_i$ (i.e., $s \xrightarrow{i}_M t$). Let $\mathcal{M}^{\mathtt{lt}}$ be the set of those KS whose accessibility relations are all left-total. Notice that for every (M,s) with $M \in \mathcal{M}^{\mathtt{lt}}$ we have $\mathcal{W}_i(M,s) \neq \emptyset$. From this observation we can prove the following.

Proposition 4.3.1 (Left-total space-consistency)**.** *The set $\mathcal{M}^{lt}$ satisfies space consistency.*

Proof. Recall that in $\mathbf{K}(\mathcal{M}^{\mathtt{lt}})$, $\mathit{false} = \emptyset$. $[\mathit{false}]_i = \{(M,s) \in \Delta \mid \forall t \text{ if } s \xrightarrow{i}_M t \text{ then } (M,t) \in \mathit{false}\} = \{(M,s) \in \Delta \mid \forall t \text{ if } s \xrightarrow{i}_M t \text{ then } (M,t) \in \emptyset\}$. Take an arbitrary $(M,s) \in \Delta$. Since the accessibility relations in $\mathbf{K}(\mathcal{M}^{\mathtt{lt}})$ are left-total, there exists t such that $s \xrightarrow{i}_M t$ but $(M,t) \notin \emptyset$. Thus for every $(M,s) \in \Delta$, $(M,s) \notin [\mathit{false}]_i$, hence $[\mathit{false}]_i = \emptyset = \mathit{false}$. □

We say that a set $\mathcal{S}$ of KS's *satisfies surjectivity* iff every space function in $\mathbf{K}(\mathcal{S})$ is surjective. The surjectivity of space functions is a necessary condition for the existence of extrusion (Corollary 3.5.1).

We can show that $\mathcal{M}^{\mathtt{lt}}$ does not satisfy surjectivity by taking $M \in \mathcal{M}^{\mathtt{lt}}$, (M,s) and (M,s') such that $s \neq s'$ and $\mathcal{W}_i(M,s) = \mathcal{W}_i(M,s')$. Let $c \in \mathcal{P}(\Delta)$ such that $c = [d]_i$ for some $d \in \mathcal{P}(\Delta)$. Since $\mathcal{W}_i(M,s) = \mathcal{W}_i(M,s')$, from Equation 4.3.1 we conclude that if $(M,s) \in c$ then $(M,s') \in c$. Thus, surjectivity is not satisfied by $[\cdot]_i$ since for every $d \in \mathcal{P}(\Delta)$, $[d]_i \neq \{(M,s)\}$. Thus $\mathbf{K}(\mathcal{M}^{\mathtt{lt}})$ cannot be extended to an scs with extrusion.

Left-unique KS's A natural general condition to prevent counter-examples to surjectivity as the one above is to restrict $\mathcal{M}^{\mathtt{lt}}$ to KS's whose accessibility relations are *left-unique*. More precisely, we say that an accessibility relation $\mathcal{R}_i$ is a *left-unique* (or *injective*) iff for every t there is *at most one* s such that $s \xrightarrow{i}_M t$. Let $\mathcal{M}^{\mathtt{ltu}}$ be the set of those KS whose accessibility relations are both left-total and left-unique. Notice that the left-unique condition guarantees that $\mathcal{W}_i(M,s) \cap \mathcal{W}_i(M,s') = \emptyset$ for any $s \neq s'$ and $M \in \mathcal{M}^{\mathtt{ltu}}$.

Proposition 4.3.2 (Left-unique surjectivity)**.** *The set $\mathcal{M}^{\mathtt{ltu}}$ satisfies surjectivity.*

Proof. It follows from Claim C.1 in the proof of Lemma 4.3.1. □

We now have an scs $\mathcal{M}^{\mathtt{ltu}}$ whose space functions are surjective. As we pointed out earlier, the Axiom of Choice implies the existence of extrusion functions (right inverses). We want, however, constructive definitions like the ones given in Section 3.4 with Theorems 3.4.1 and 3.4.3.

We cannot apply Theorem 3.4.3 because the spatial functions of $\mathcal{M}^{\mathtt{ltu}}$ are not meet-complete. For a counter-example take (M, s) with $\mathcal{W}_i(M, s) = \{t, u\}$. Recall that the meet $\sqcap$ in $\mathbf{K}(\mathcal{M}^{\mathtt{ltu}})$ is set union. One can verify that $\{(M, s)\} = [\ \{(M, t), (M, u)\}\ ]_i \neq [\ \{(M, t)\}\ ]_i \cup [\ \{(M, u)\}\ ]_i = \emptyset$.

Nevertheless, the space functions of any Kripke scs are *continuous*.

Proposition 4.3.3. *The space functions of $\boldsymbol{K}(\mathcal{M})$ are continuous.*

Proof. It suffices to show that space functions are join-complete (this implies continuity): i.e., that for every agent i and every set $S \subseteq \mathcal{P}(\Delta)$ we have $[\bigcap S]_i = \bigcap [S]_i$. By definition $(M, s) \in \bigcap S \Leftrightarrow \forall_{c \in S}\ (M, s) \in c$. We can then conclude

$$\{(M, s) \in \Delta \mid \forall t \text{ if } s \xrightarrow{i}_M t \text{ then } (M, t) \in \bigcap S\} =$$
$$\bigcap_{c \in S} \{(M, s) \in \Delta \mid \forall t \text{ if } s \xrightarrow{i}_M t \text{ then } (M, t) \in c\}$$

as wanted. □

Therefore we can apply Theorem 3.4.1 and derive the following extrusion function for each $[\cdot]_i$ in $\mathbf{K}(\mathcal{M}^{\mathtt{ltu}})$:

$$\uparrow_i : c \mapsto \bigsqcup [c]_i^{-1}. \tag{4.3.2}$$

Furthermore, we can show that the construction in Equation 4.3.2 is equivalent to the intensional definition given below.

Lemma 4.3.1. *(Extrusion for Kripke Spaces). Let $\uparrow_i$ be defined as in Equation 4.3.2 over the Kripke scs $\boldsymbol{K}(\mathcal{M}^{\mathtt{ltu}})$. Then*

$$\uparrow_i(c) = i^{-1}(c) \text{ with } i^{-1}(c) \stackrel{\mathtt{def}}{=} \{(M, t) \in \Delta \mid \exists s :\ s \xrightarrow{i}_M t \text{ and } (M, s) \in c\} \tag{4.3.3}$$

where Δ *is the set of pointed KS* (M, s) *such that* $M \in \mathcal{M}^{\mathtt{ltu}}$ *and* s *is a state of* M*.*

Proof. Let

$$\uparrow'_i c = \{(M,t) \in \Delta \mid \exists s :\ s \xrightarrow{i}_M t \text{ and } (M,s) \in c\}. \tag{4.3.4}$$

We need to prove $\uparrow'_i c = \uparrow_i c$ where $\uparrow_i c = \bigsqcup [c]_i^{-1} = \bigcap [c]_i^{-1}$. Recall the definition of $[\cdot]_i$:

$$[c]_i = \{(M,s) \in \Delta \mid \forall t \text{ if } s \xrightarrow{i}_M t \text{ then } (M,t) \in c\}.$$

— $\uparrow'_i c \subseteq \uparrow_i c$. Suppose that (H.1) $(M,t) \in \uparrow'_i c$ but (H.2) $(M,t) \notin \uparrow_i c$. From H.1, Equation 4.3.4, and the fact that accessibility relations in $\mathbf{K}(\mathcal{M}^{\mathtt{ltu}})$ are left-unique, there exists a unique state, let us call it s in what follows, such that $s \xrightarrow{i}_M t$ and $(M,s) \in c$. From H.2 we know that there exists some $d \in [c]_i^{-1}$ such that $(M,t) \notin d$. Notice that $[d]_i = c$ since $d \in [c]_i^{-1}$. But since $(M,t) \notin d$ and $s \xrightarrow{i}_M t$ we conclude $(M,s) \notin [d]_i$. This is a contradiction since we previously concluded $[d]_i = c$ and $(M,s) \in c$.

— $\uparrow_i c \subseteq \uparrow'_i c$. We claim that (C.1) $[\uparrow'_i c]_i = c$. From this claim we have $\uparrow'_i c \in [c]_i^{-1}$, therefore $\uparrow_i c = \bigcap [c]_i^{-1} \subseteq \uparrow'_i c$ as wanted.

It remains to prove **C**laim C.1: $[\uparrow'_i c]_i = c$ for every $c \in \mathcal{P}(\mathcal{M}^{\mathtt{ltu}})$.

— Assume (H.3) $(M,s) \in c$, we wish to prove $(M,s) \in [\uparrow'_i c]_i$. From the definition of $[\cdot]_i$, we need to show that (H.4) for any t if $s \xrightarrow{i}_M t$ then $(M,t) \in c$. From H.3, H.4 and Equation 4.3.4 we obtain $(M,s) \in [\uparrow'_i c]_i$ as we wished.

— Assume (H.5) $(M,s) \in [\uparrow'_i c]_i$, we want to prove $(M,s) \in c$. Since the accessibility relations in $\mathbf{K}(\mathcal{M}^{\mathtt{ltu}})$ are left-total, from H.5 we conclude that there exists t such that $s \xrightarrow{i}_M t$ and $(M,t) \in \uparrow'_i c$. From $(M,t) \in \uparrow'_i c$ it follows that there exists s' such that $s' \xrightarrow{i}_M t$ and $(M,s') \in c$. From the fact that the accessibility relations in $\mathbf{K}(\mathcal{M}^{\mathtt{ltu}})$ are left-unique we conclude that $s = s'$, and thus $(M,s) \in c$.

□

From the above we can now extend $\mathbf{K}(\mathcal{M}^{\mathtt{ltu}})$ to the following scs with extrusion.

Definition 4.3.1 (Kripke scs with extrusion)**.** *The n-agent scs with extrusion* $\boldsymbol{K}^{\uparrow}(\mathcal{M}^{\mathtt{ltu}})$ *results from extending* $\boldsymbol{K}(\mathcal{M}^{\mathtt{ltu}})$ *with an extrusion function* $\uparrow_i$ *for each* $i \in \{1, \ldots, n\}$ *defined as in Equation 4.3.2.*

It follows from Property 3.6.1 that in the derived scse, space and extrusion form a *Galois connection.*

Corollary 4.3.1. *Let* $[\cdot]_i$ *and* $\uparrow_i$ *be the space and extrusion function of agent* i *in* $\boldsymbol{K}^{\uparrow}(\mathcal{M}^{\mathtt{ltu}})$*. The pair* $([\cdot]_i, \uparrow_i)$ *is a Galois connection.*

We shall apply this Galois connection in the following section.

The BU_n logic

We shall now extend the modal language in Definition 4.1.1 with modalities to express utterances. The intended meaning and properties of the formulae in the extended language will be given from the scse $\mathbf{K}^{\uparrow}(\mathcal{M}^{\mathtt{ltu}})$ we derived in the previous section (Definition 4.3.1). We shall refer to the resulting multi-modal logic as BU_n.

For clarity we shall write B_i instead of $\Box_i$. The language $\mathcal{L}_n^{BU}(\Phi)$ is obtained by replacing $\Box_i$ with B_i in the grammar of Example 4.2.1 and extending it with modalities U_i.

Definition 4.3.2 (Modal language for utterance)**.** *Let* $\mathcal{L}_n^{BU}(\Phi)$ *with* $n \geq 1$ *be the language built from a set of primitive propositions* Φ *by the following syntax:*

$$\varphi \;:=\; p \mid \varphi \wedge \varphi \mid \neg\varphi \mid B_i\varphi \mid U_i\varphi$$

where $i \in \{1 \ldots n\}$ *and* $p \in \Phi$*.*

Before presenting the semantics of BU_n, we give a dual spatial/epistemic intuition about its modal formulae. Let us consider s and t such that $s \xrightarrow{i}_M t$. Recall from Notation 4.1.2 that t is a world that agent i considers possible in the world s (of a KS M). In spatial terms we can think of t as being a *local world* for agent i wrt to the *outside world* s. If the *belief* modality $B_i\varphi$ holds true in outside world s it implies that φ must be true in the local world t.

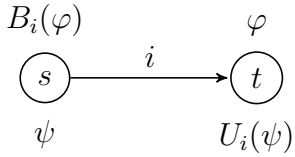


Figure 4.1 – Illustration of $s \xrightarrow{i}_M t$ with $B_i\varphi$ and ψ true at s and φ and $U_i\psi$ true at t

Similarly, if the utterance modality $U_i\psi$ holds in the local world t it implies that ψ must be true in the outside world s. Figure 4.1 illustrates the above-mentioned description.

Derived Specifications We expect the following formula to be valid:

$$B_iU_i\varphi \Leftrightarrow \varphi. \tag{4.3.5}$$

The above equation can be seen as agent i uttering φ. We can also derive specifications for common social behaviors such as:

$$\mathcal{O}_i(\varphi) \stackrel{\texttt{def}}{=} B_i(\varphi \wedge U_i(\varphi)) \text{ and } \mathcal{H}_i(\varphi) \stackrel{\texttt{def}}{=} B_i(\neg\varphi \wedge U_i(\varphi)).$$

An *opinion* $\mathcal{O}_i(\varphi)$ by agent i is the utterance of a statement φ that the agent believes true. Thus we expect the validity of the following:

$$\mathcal{O}_i(\varphi) \Leftrightarrow (B_i\varphi) \wedge \varphi. \tag{4.3.6}$$

A *hoax* or *intentional lie* $\mathcal{H}_i(\varphi)$ by agent i is the utterance of a statement φ that the agent believes false: Thus

$$\mathcal{H}_i(\varphi) \Leftrightarrow (B_i\neg\varphi) \wedge \varphi. \tag{4.3.7}$$

should be valid. We also define duals of belief and utterance as:

$$\hat{B}_i\varphi \stackrel{\texttt{def}}{=} \neg B_i\neg\varphi \text{ and } \hat{U}_i\varphi \stackrel{\texttt{def}}{=} \neg U_i\neg\varphi.$$

The formula $\hat{B}_i\varphi$ states that φ is consistent with agent i's beliefs. Similarly $\hat{U}_i\varphi$ means φ is consistent with agent i's utterances. We expect the validity of the following formulae:

$$B_i\varphi \Rightarrow \hat{B}_i\varphi \text{ and } U_i\varphi \Rightarrow \hat{U}_i\varphi. \tag{4.3.8}$$

The formulae in Equation 4.3.8 are consistency axioms. The first formula says that if agent i believes φ then it should not believe $\neg\varphi$. The other says that the extrusion of φ and $\neg\varphi$ would generate an inconsistency.

Semantics We now give the semantics for BU_n using the scse $\mathbf{K}^{\uparrow}(\mathcal{M}^{\mathtt{ltu}})$ in Definition 4.3.1. Recall our definition of the negation constraint $\sim c$ (Definition 3.2.1) and that $\mathbf{K}^{\uparrow}(\mathcal{M}^{\mathtt{ltu}})$ is also a frame (Remark 3.2.1).

Definition 4.3.3. *Let $\boldsymbol{K}^{\uparrow}(\mathcal{M}^{ltu}) = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n, \uparrow_1, \ldots, \uparrow_n)$ be the scse in Definition 4.3.1. Given φ in $\mathcal{L}_n^{BU}(\Phi)$, its denotation $\boldsymbol{K}^{\uparrow}[\![\varphi]\!]$ is inductively defined as follows:*

$$\begin{aligned}
\boldsymbol{K}^{\uparrow}[\![p]\!] &= \{(M,s) \in \Delta |\ \pi_M(s)(p) = 1\} \\
\boldsymbol{K}^{\uparrow}[\![\varphi \wedge \varphi']\!] &= \boldsymbol{K}^{\uparrow}[\![\varphi]\!] \sqcup \boldsymbol{K}^{\uparrow}[\![\varphi']\!] \\
\boldsymbol{K}^{\uparrow}[\![\neg\varphi]\!] &= \sim \boldsymbol{K}^{\uparrow}[\![\varphi]\!] \\
\boldsymbol{K}^{\uparrow}[\![B_i\varphi]\!] &= [\ \boldsymbol{K}^{\uparrow}[\![\varphi]\!]\ ]_i \\
\boldsymbol{K}^{\uparrow}[\![U_i\varphi]\!] &= \uparrow_i \boldsymbol{K}^{\uparrow}[\![\varphi]\!]
\end{aligned}$$

where Δ is the set of all pointed Kripke structures (M,s) such that $M \in \mathcal{M}^{ltu}$. We say that φ is valid in BU_n iff $\boldsymbol{K}^{\uparrow}[\![\varphi]\!] = true$.

From the above semantics definition and the properties of scs with extrusion one can verify the expected behavior of utterance, opinion, and hoaxes in Equations 4.3.5, 4.3.6, 4.3.7 and 4.3.8.

Proposition 4.3.4. *The formulae in Equations 4.3.5, 4.3.6, 4.3.7 and 4.3.8 are valid in BU_n.*

Proof. We show the validity of the various formulae using their semantics definition and the axioms of space and extrusion.

— $B_iU_i\varphi \Leftrightarrow \varphi$ is valid.
We need to prove $\mathbf{K}^{\uparrow}[\![B_iU_i\varphi]\!] = \mathbf{K}^{\uparrow}[\![\varphi]\!]$. We use the semantic definitions along with E.1 on the left side to obtain $\mathbf{K}^{\uparrow}[\![B_iU_i\varphi]\!] = [\ \mathbf{K}^{\uparrow}[\![U_i\varphi]\!]\ ]_i = [\ \uparrow_i \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i = \mathbf{K}^{\uparrow}[\![\varphi]\!]$

— $\mathcal{O}_i(\varphi) \Leftrightarrow B_i\varphi \wedge \varphi$ is valid.
We need to prove $\mathbf{K}^{\uparrow}[\![B_i(\varphi \wedge U_i(\varphi))]\!] = \mathbf{K}^{\uparrow}[\![B_i\varphi \wedge \varphi]\!]$. We use the semantic definitions on the left side to obtain $\mathbf{K}^{\uparrow}[\![B_i(\varphi \wedge U_i(\varphi))]\!] = [\ \mathbf{K}^{\uparrow}[\![\varphi \wedge U_i(\varphi)]\!]\ ]_i = [\ \mathbf{K}^{\uparrow}[\![\varphi]\!]\ \sqcup\ \mathbf{K}^{\uparrow}[\![U_i(\varphi)]\!]\ ]_i$. Using S.2 and E.1 we obtain $[\ \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i \sqcup [\ \mathbf{K}^{\uparrow}[\![U_i(\varphi)]\!]\ ]_i = [\ \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i \sqcup [\ \uparrow_i \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i = [\ \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i \sqcup \mathbf{K}^{\uparrow}[\![\varphi]\!] = \mathbf{K}^{\uparrow}[\![B_i\varphi]\!] \sqcup \mathbf{K}^{\uparrow}[\![\varphi]\!] = \mathbf{K}^{\uparrow}[\![B_i\varphi \wedge \varphi]\!]$.

— $\mathcal{H}_i(\varphi) \Leftrightarrow B_i\neg\varphi \wedge \varphi$ is valid.
We need to prove $\mathbf{K}^{\uparrow}[\![B_i(\neg\varphi \wedge U_i(\varphi))]\!] = \mathbf{K}^{\uparrow}[\![B_i\neg\varphi \wedge \varphi]\!]$. We use the semantic definitions on the left side to obtain $\mathbf{K}^{\uparrow}[\![B_i(\neg\varphi \wedge U_i(\varphi))]\!] = [\ \mathbf{K}^{\uparrow}[\![\neg\varphi \wedge U_i(\varphi)]\!]\ ]_i = [\ \mathbf{K}^{\uparrow}[\![\neg\varphi]\!]\ \sqcup\ \mathbf{K}^{\uparrow}[\![U_i(\varphi)]\!]\ ]_i$. Using S.2 and E.1 we obtain $[\ \mathbf{K}^{\uparrow}[\![\neg\varphi]\!]\ ]_i \sqcup [\ \mathbf{K}^{\uparrow}[\![U_i(\varphi)]\!]\ ]_i = [\ \mathbf{K}^{\uparrow}[\![\neg\varphi]\!]\ ]_i \sqcup [\ \uparrow_i \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i = [\ \mathbf{K}^{\uparrow}[\![\neg\varphi]\!]\ ]_i \sqcup \mathbf{K}^{\uparrow}[\![\varphi]\!] = \mathbf{K}^{\uparrow}[\![B_i\neg\varphi]\!] \sqcup \mathbf{K}^{\uparrow}[\![\varphi]\!] = \mathbf{K}^{\uparrow}[\![B_i\neg\varphi \wedge \varphi]\!]$.

— $B_i\varphi \Rightarrow \hat{B}_i\varphi$ is valid.
It suffices to prove $\mathbf{K}^{\uparrow}[\![\hat{B}_i\varphi]\!] = \mathbf{K}^{\uparrow}[\![\neg B_i\neg\varphi]\!] = {\sim}\mathbf{K}^{\uparrow}[\![B_i\neg\varphi]\!] = {\sim}[\ \mathbf{K}^{\uparrow}[\![\neg\varphi]\!]\ ]_i = {\sim}[{\sim}\mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i \sqsubseteq \mathbf{K}^{\uparrow}[\![B_i\varphi]\!] = [\ \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i$. Notice that $[\ \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i$ is defined as

$$\{(M,s) \in \Delta \mid \forall t \text{ if } s \xrightarrow{i}_M t \text{ then } \ (M,t) \in \mathbf{K}^{\uparrow}[\![\varphi]\!]\}.$$

Similarly, ${\sim}[{\sim}\mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i$ is equivalent to the set

$$\{(M,s) \in \Delta \mid \exists t : s \xrightarrow{i}_M t \text{ and } (M,t) \in \mathbf{K}^{\uparrow}[\![\varphi]\!]\}.$$

Using these equations we can verify that $[\ \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i \subseteq {\sim}[{\sim}\mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i$. Therefore ${\sim}[{\sim}\mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i \sqsubseteq [\ \mathbf{K}^{\uparrow}[\![\varphi]\!]\ ]_i$ as wanted.

— $U_i\varphi \Rightarrow \hat{U}_i\varphi$ is valid.
Analogous to the previous case.

□

Notice that $\varphi \Rightarrow \psi$ is valid in BU_n iff $\mathbf{K}^{\uparrow}[\![\psi]\!] \sqsubseteq \mathbf{K}^{\uparrow}[\![\varphi]\!]$. It follows from Corollary 4.3.1 that in $\mathbf{K}^{\uparrow}(\mathcal{M}^{\mathtt{ltu}})$ we have $[c]_i \sqsubseteq d$ iff $c \sqsubseteq \uparrow_i(d)$. We can then conclude the following property.

Corollary 4.3.2. *$\varphi \Rightarrow B_i\psi$ is valid in BU_n iff $U_i\varphi \Rightarrow \psi$ is valid in BU_n.*

Intuitively Corollary 4.3.2 says that belief and utterance form a Galois connection. We can therefore reduce the validity of the implication of a belief property to/from the implication by a utterance property.

We conclude this chapter by revisiting Example 3.2.2.

Example 4.3.1 (Hoax Communication)**.** *Let us consider the epistemic formulae* $F = B_j(\neg p \Rightarrow U_j(q))$ *and* $G = B_i(p \wedge U_i B_j(\neg p))$*. The formula* F *specifies an agent* j *that would utter* q *if she believed that* p *was not true. The formula* G *specifies an agent* i *that believes* p *and yet he utters that* j *should believe the opposite, i.e., a lie* $\neg p$ *by agent* i*. We can show that the combination of both specifications implies that* q *will be extruded, i.e., we will show that*

$$(F \wedge G) \Rightarrow q$$

is valid in BU_n*. We obtain the following derivation using the properties of the space and extrusion functions:*

$$\begin{aligned}
\boldsymbol{K}^{\uparrow}[\![F \wedge G]\!] &= [\sim \boldsymbol{K}^{\uparrow}[\![p]\!] \to \uparrow_j \boldsymbol{K}^{\uparrow}[\![q]\!]]_j \sqcup [\boldsymbol{K}^{\uparrow}[\![p]\!] \sqcup \uparrow_i [\sim \boldsymbol{K}^{\uparrow}[\![p]\!]]_j]_i && \textit{Def.4.3.3} \\
&= [\sim \boldsymbol{K}^{\uparrow}[\![p]\!] \to \uparrow_j \boldsymbol{K}^{\uparrow}[\![q]\!]]_j \sqcup [\boldsymbol{K}^{\uparrow}[\![p]\!]]_i \sqcup [\uparrow_i [\sim \boldsymbol{K}^{\uparrow}[\![p]\!]]_j]_i && \textit{S.2}\ [\cdot]_i \\
&= [\sim \boldsymbol{K}^{\uparrow}[\![p]\!] \to \uparrow_j \boldsymbol{K}^{\uparrow}[\![q]\!]]_j \sqcup [\boldsymbol{K}^{\uparrow}[\![p]\!]]_i \sqcup [\sim \boldsymbol{K}^{\uparrow}[\![p]\!]]_j && \textit{E.1}\ [\cdot]_i \\
&= [\sim \boldsymbol{K}^{\uparrow}[\![p]\!] \sqcup \sim \boldsymbol{K}^{\uparrow}[\![p]\!] \to \uparrow_j \boldsymbol{K}^{\uparrow}[\![q]\!]]_j \sqcup [\boldsymbol{K}^{\uparrow}[\![p]\!]]_i && \textit{S.2}\ [\cdot]_j \\
&= [\sim \boldsymbol{K}^{\uparrow}[\![p]\!] \sqcup \uparrow_j \boldsymbol{K}^{\uparrow}[\![q]\!]]_j \sqcup [\boldsymbol{K}^{\uparrow}[\![p]\!]]_i && \textit{Lem.3.2.1} \\
&= \boldsymbol{K}^{\uparrow}[\![q]\!] \sqcup [\sim \boldsymbol{K}^{\uparrow}[\![p]\!]]_j \sqcup [\boldsymbol{K}^{\uparrow}[\![p]\!]]_i && \textit{E.1}\ [\cdot]_j \\
&= \boldsymbol{K}^{\uparrow}[\![q]\!] \cap [\sim \boldsymbol{K}^{\uparrow}[\![p]\!]]_j \cap [\boldsymbol{K}^{\uparrow}[\![p]\!]]_i && \textit{Def.4.2.1} \\
&\subseteq \boldsymbol{K}^{\uparrow}[\![q]\!]
\end{aligned}$$

Thus every model of $\boldsymbol{K}^{\uparrow}[\![F \wedge G]\!]$ *is a model of* $\boldsymbol{K}^{\uparrow}[\![q]\!]$*. We can conclude that* $(F \wedge G) \Rightarrow q$ *is valid as wanted.* □

4.4 Summary

After introducing Kripke constraint systems, we devised a bimdodal logic of belief and utterance where we were able to define epistemic behaviors such as

opinions and hoaxes. We then proceeded to identify the necessary condition for the Kripke cs to be able to derive extrusion functions using Theorem 3.4.1. Subsequently, we characterized formulae of the bimodal logic with semantics using the Kripke cs with extrusion. In this way we were able to provide a semantic represenation of the definition of opinions and hoaxes.

The work exposed in this chapter is contained in [GPRV16, HPRV15]. The following chapter explains how to represent a stricter epistemic behavior, that of knowledge.

Chapter 5

Knowledge in Terms of Global Space

In the previous chapter we saw how spatial constraint systems can be used to represent epistemic concepts such as beliefs, lies and opinions. In this chapter we show that spatial constraint systems can also express the epistemic concept of knowledge using the notion of global information from Definition 2.2.3. We begin with an introduction to epistemic logic and the different axioms attributed to knowledge.

5.1 Epistemic Logic

When conceiving a modal logic, one can impose an interpretation on the modal operator for contextualizing its application. One usch important interpretation is that of *knowledge* wherein the logic takes the name of *Epistemic Logic*.

Similarly to what we did in Section 4.3, we now write the modal operator as K. Thus, $K_i\phi$ can be read as "agent i knows ϕ" and nesting of modalities like $K_iK_j\phi$ is read as "agent i knows that agent j knows ϕ". An introductory text on the theory and application of epistemic logics is [FHMV95].

In epistemic logics, the axiom system $\mathbf{K}_n$ (Example 4.1.1) has a more contextualized interpretation. Axiom A.2 is referred to as the Knowledge distribution axiom, intuitively speaking this means agents are perfect reasoners. Rule R.2 is called Knowledge generalization, meaning that universal truths are known by all the agents.

Axiom	**Property of $\mathcal{R}_i$**
A.3 $K_i\phi \Rightarrow \phi$	reflexive
A.4 $K_i\phi \Rightarrow K_iK_i\phi$	transitive
A.5 $\neg K_i\phi \Rightarrow K_i\neg K_i\phi$	euclidian
A.6 $\neg K_i(\mathit{false})$	serial
A.7 $\phi \Rightarrow K_i\neg K_i\neg\phi$	symmetric

Table 5.1 – Axiom correspondence to possibility relation

Nonetheless, this axiom system alone usually does not capture properties frequently imputed by philosopher to knowledge. We list some of these common properties of knowledge as axioms.

For all agents $i = 1 \ldots n$:

A.3 $K_i\phi \Rightarrow \phi$ (Knowledge Axiom)

A.4 $K_i\phi \Rightarrow K_iK_i\phi$ (Positive Introspection Axiom)

A.5 $\neg K_i\phi \Rightarrow K_i\neg K_i\phi$ (Negative Introspection Axiom)

A.6 $\neg K_i(\mathit{false})$ (Consistency Axiom)

Axiom A.3 captures the notion of knowledge, if agent i knows something then it is true . In other words, agents have factual knowledge, they cannot know something that is untrue. This might not be the normal case, specially if the modality is interpreted as belief instead of knowledge. In this case A.3 should be replaced with A.6, i.e. the agent's set of beliefs are always consistent. Subsequently, we could improve readability by writing the modality as B (e.g. A.6 $\neg B_i(\mathit{false})$).

Some important axioms systems are combinations of these properties. Axiom system **K** together with axiom A.3 or axiom A.6 are called **T** and **D** respectively. Axiom system **T** with axiom A.4 and also axiom A.5 are called S4 and S5 respectively. Adding axiom A.4 and A.5 to **D** is called axiom system KD45.

Semantically speaking, all axioms in an axiom system must be valid wrt a certain class of KS. The axiom system **K** is valid wrt the class of KS whose possibility relations are unrestricted (written as $\mathcal{M}$). Table 5.1 shows the correspondence between axioms and the properties of the possibility relation.

Axiom A.7 $\phi \Rightarrow K_i\neg K_i\neg\phi$ is a consequence of axioms A.3 and A.5. Going by Notation 4.1.2, for all $s, t, u \in S$ we have that $\mathcal{R}_i$ is; — *reflexive* if $(s, s) \in \mathcal{R}_i$, — *transitive* whenever $(s, t) \in \mathcal{R}_i$ and $(t, u) \in \mathcal{R}_i$ then $(s, u) \in \mathcal{R}_i$, *euclidian*

if $(s,t) \in \mathcal{R}_i$ and (s,u) imply $(t,u) \in \mathcal{R}_i$, and — *serial* if there exists s' s.t. $(s,s') \in \mathcal{R}_i$.

Notice that S5 (i.e. Axioms A.1 - A.5 and by consequence A.7) requires the accessibility relation to be reflexive, transitive and symmetric. In other words, the axiom system S5 requires the accessibility relations to be equivalence relations.

5.2 Knowledge Constraint System

In [KPPV12] the authors extended the notion of spatial constraint system to account for *knowledge*. In this chapter we shall refer to the extended notion in [KPPV12] as *S4 constraint systems* since it is meant to capture the epistemic logic for knowledge S4. Roughly speaking, one may wish to use $[c]_i$ to represent not only some information c that agent i has but rather a *fact* that he knows. The domain theoretical nature of constraint systems allows for a rather simple and elegant characterization of knowledge by requiring space functions to be *Kuratowski closure operators* [MT44]: i.e., monotone, extensive and idempotent functions that preserve bottom and lubs.

Definition 5.2.1 (Knowledge Constraint System [KPPV12])**.** *An n-agent* S4 constraint system (n-s4cs) $\boldsymbol{C}$ *is an n-scs whose space functions $[\cdot]_1, \ldots, [\cdot]_n$ are also* closure operators*. Thus, in addition to S.1 and S.2 in Definition 2.2.2, each $[\cdot]_i$ also satisfies: (EP.1) $[c]_i \sqsupseteq c$ and (EP.2) $[[c]_i]_i = [c]_i$.*

Intuitively, in an n-s4cs, $[c]_i$ states that the agent i has knowledge of c in its store $[\cdot]_i$. The axiom EP.1 says that if agent i knows c then c must hold, hence $[c]_i$ has at least as much information as c. The epistemic principle that an agent i is aware of its own knowledge (*the agent knows what he knows*) is realized by EP.2. Also the epistemic assumption that agents are *idealized reasoners* follows from the monotonicity of space functions (Remark 2.2.2); for if c is a consequence of d ($d \sqsupseteq c$) then if d is known to agent i, so is c, $[d]_i \sqsupseteq [c]_i$.

Recall that modal logics are interpreted over families of Kripke structures obtained by restricting their accessibility relations. Also, $\mathcal{M}_n(\Phi)$ is used to denote the set of *all* Kripke structures over the set of primitive propositions Φ (Notation 4.3.1). We shall use $\mathcal{M}_n^{\mathtt{rt}}(\Phi)$ to denote the set of those n-agents Kripke structures,

over the set of primitive propositions Φ, whose accessibility relations are *reflexive* and *transitive*. As in the previous chapter, for notational convenience, we take the set Φ of primitive propositions and n to be fixed from now on and omit them often from the notation. E.g., we write $\mathcal{M}^{\mathtt{rt}}$ instead of $\mathcal{M}_n^{\mathtt{rt}}(\Phi)$.

Henceforth we use $\mathbf{C}^{\mathtt{rt}}$ to denote the Kripke constraint system $\mathbf{K}(\mathcal{M}^{\mathtt{rt}})$ (Definition 4.2.1). In [KPPV12] it was shown that $\mathbf{C}^{\mathtt{rt}}$ is in fact an S4 constraint system.

Proposition 5.2.1 ([KPPV12])**.** $\boldsymbol{C}^{\mathtt{rt}}$ *is an* s4cs.

Recall our interpretation of formulae of the modal language $\mathcal{L}_n(\Phi)$ (Definition 4.1.1) using Kripke spatial constraint systems (Definition 4.2.2). In particular the interpretation of the formula $\Box_i(\phi)$ in $\mathbf{C}^{\mathtt{rt}}$, denoted $\mathbf{C}^{\mathtt{rt}}[\![\Box_i(\phi)]\!]$, is the constraint $[\mathbf{C}^{\mathtt{rt}}[\![\phi]\!]]_i$. Let us now recall the notion of validity in the modal logic S4 [Hin62].

Definition 5.2.2. *Let ϕ be a modal formula from the modal language $\mathcal{L}_n(\Phi)$. The formula ϕ is said to be* S4-valid *iff for every (M, s) where $M \in \mathcal{M}^{\mathtt{rt}}(\Phi)$ and s is a state of M, we have $(M, s) \models \phi$.*

Notation 5.2.1. *As stated at the beginning of this chapter, in the modal logic S4 the box modality $\Box_i$ is often written as K_i. The formula $K_i(\phi)$ specifies that agent i knows ϕ.*

The following proposition from [KPPV12] is an immediate consequence of the above definition. It states the correctness wrt validity of the interpretation of S4 formulae in $\mathbf{C}^{\mathtt{rt}}$.

Proposition 5.2.2 ([KPPV12])**.** $\boldsymbol{C}^{\mathtt{rt}}[\![\phi]\!] =$ *true if and only if ϕ is S4-valid.*

The above gives a brief summary of the use in [KPPV12] of *Kuratowski closure operators* $[c]_i$ to capture knowledge. In what follows we show an alternative interpretation of knowledge as the global construct $[\![c]\!]_G$ in Definition 2.2.3.

5.3 Knowledge as Global Information

Let $\mathcal{C} = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$ be a *spatial* constraint system. From Definition 2.2.3 we obtain the following equation:

$$[\![c]\!]_{\{i\}} = c \sqcup [c]_i \sqcup [[c]_i]_i \sqcup [[[c]_i]_i]_i \sqcup \ldots = c \sqcup [c]_i \sqcup [c]_i^2 \sqcup [c]_i^3 \sqcup \ldots = \bigsqcup_{j=0}^{\infty} [c]_i^j \quad (5.3.1)$$

where $[e]_i^0 \stackrel{\texttt{def}}{=} e$ and $[e]_i^{j+1} \stackrel{\texttt{def}}{=} [\ [e]_i^j\ ]_i$. For simplicity we shall use $[\![\cdot]\!]_i$ as an abbreviation of $[\![\cdot]\!]_{\{i\}}$.

Intuitively, $[\![c]\!]_i$ says that c holds *globally* wrt i: c holds outside and in every nested space of agent i. We shall demonstrate that $[\![c]\!]_i$ can also be used to represent the knowledge of c by agent i.

We will show that the global function $[\![c]\!]_i$ is in fact a Kuratowski closure operator and thus satisfies the epistemic axioms EP.1 and EP.2 above: It is easy to see that $[\![c]\!]_i$ satisfies $[\![c]\!]_i \sqsupseteq c$ (EP.1). Under certain natural assumption we shall see that it also satisfies $[\![[\![c]\!]_i]\!]_i = [\![c]\!]_i$ (EP.2). Furthermore, we can combine knowledge with our belief interpretation of space functions: Clearly, $[\![c]\!]_i \sqsupseteq [c]_i$ holds for any c. This reflects the epistemic principle that *whatever is known is also believed* [Hin62].

We now show that any *spatial constraint system* with continuous space functions $[\cdot]_1, \ldots, [\cdot]_n$ induces an s4cs with space functions $[\![\cdot]\!]_1, \ldots [\![\cdot]\!]_n$.

Definition 5.3.1. *Given an scs $\mathcal{C} = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$, we use $\mathcal{C}^*$ to denote the tuple $(Con, \sqsubseteq, [\![\cdot]\!]_1, \ldots, [\![\cdot]\!]_n)$.*

One can show that $\mathcal{C}^*$ is also a spatial constraint system. Furthermore it is an s4cs as stated next.

Theorem 5.3.1. *Let $\mathcal{C} = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$ be a spatial constraint system. If $[\cdot]_1, \ldots, [\cdot]_n$ are continuous functions then $\mathcal{C}^*$ is an n-agent s4cs.*

Proof. We need to show that each $[\![\cdot]\!]_i$ satisfies S.1, S.2, EP.1 and EP.2.

— We want to prove that $[\![\cdot]\!]_i$ satisfies S.1: $[\![\mathit{true}]\!]_i = \mathit{true}$. Since $[\cdot]_i$ satisfies S.1 we can use $[\mathit{true}]_i = \mathit{true}$ to show, by induction on j, that $[\mathit{true}]_i^j = \mathit{true}$ for any j. Then from Equation 5.3.1 we conclude $[\![\mathit{true}]\!]_i = \bigsqcup\{\mathit{true}\} = \mathit{true}$ as wanted.

— We want to prove that $[\![\cdot]\!]_i$ satisfies S.2: $[\![c \sqcup d]\!]_i = [\![c]\!]_i \sqcup [\![d]\!]_i$. Since $[\cdot]_i$ satisfies S.2 we can use $[c \sqcup d]_i = [c]_i \sqcup [d]_i$ to show by induction on j that $[c \sqcup d]_i^j = [c]_i^j \sqcup [d]_i^j$ for any j. We then obtain the following equation

$$[\![c \sqcup d]\!]_i = \bigsqcup_{j=0}^{\infty} [c \sqcup d]_i^j = \bigsqcup_{j=0}^{\infty} ([c]_i^j \sqcup [d]_i^j). \tag{5.3.2}$$

Clearly $[\![c]\!]_i = \bigsqcup_{j=0}^{\infty} [c]_i^j \sqsubseteq \bigsqcup_{j=0}^{\infty} ([c]_i^j \sqcup [d]_i^j) \sqsupseteq \bigsqcup_{j=0}^{\infty} [d]_i^j = [\![d]\!]_i$. Therefore $[\![c \sqcup d]\!]_i \sqsupseteq [\![c]\!]_i \sqcup [\![d]\!]_i$. It remains to prove $[\![c]\!]_i \sqcup [\![d]\!]_i \sqsupseteq [\![c \sqcup d]\!]_i$.
Notice that $[\![c]\!]_i \sqcup [\![d]\!]_i = (\bigsqcup_{j=0}^{\infty} [c]_i^j) \sqcup (\bigsqcup_{j=0}^{\infty} [d]_i^j)$ is an upper bound of the set $S = \{[c]_i^j \sqcup [d]_i^j \mid j \geq 0\}$. Therefore $[\![c]\!]_i \sqcup [\![d]\!]_i \sqsupseteq \bigsqcup S = \bigsqcup_{j=0}^{\infty} ([c]_i^j \sqcup [d]_i^j) = [\![c \sqcup d]\!]_i$ as wanted.

— We want to prove that $[\![\cdot]\!]_i$ satisfies EP.1: $[\![c]\!]_i \sqsupseteq c$. Immediate consequence of Equation 5.3.1.

— Finally we prove that $[\![\cdot]\!]_i$ satisfies EP.2: $[\![[\![c]\!]_i]\!]_i = [\![c]\!]_i$. Since $[\![\cdot]\!]_i$ satisfies EP.1 we have $[\![[\![c]\!]_i]\!]_i \sqsupseteq [\![c]\!]_i$. It remains to prove that $[\![c]\!]_i \sqsupseteq [\![[\![c]\!]_i]\!]_i$.
Let $S = \{[c]_i^k \mid k \geq 0\}$. Notice that $[\![c]\!]_i = \bigsqcup S$. One can verify from the definition of $[\cdot]_i^j$ that for any j

$$[S]_i^j = \{[[c]_i^k]_i^j \mid k \geq 0\} = \{[c]_i^{k+j} \mid k \geq 0\} \subseteq S$$

From the continuity of $[\cdot]_i$, one can show by induction on j the continuity of $[\cdot]_i^j$ for any j. It then follows that $[\bigsqcup S]_i^j = \bigsqcup [S]_i^j$ for any j. Since for any j, $[S]_i^j \subseteq S$ we conclude that for every j, $\bigsqcup S \sqsupseteq [S]_i^j$. Therefore $\bigsqcup S \sqsupseteq \bigsqcup_{j=0}^{\infty} [S]_i^j$. This concludes the proof since from Equation 5.3.1 $[\![c]\!]_i = \bigsqcup S$ and $[\![[\![c]\!]_i]\!]_i = \bigsqcup_{j=0}^{\infty} [\bigsqcup S]_i^j = \bigsqcup_{j=0}^{\infty} [S]_i^j$.

□

We shall now prove that S4 can also be captured using the global interpretation of space.

From now on $\mathbf{C}$ denotes the Kripke constraint system $\mathbf{K}(\mathcal{M})$ (Definition 4.2.1). Notice that unlike in $\mathbf{C}^{\mathtt{rt}}$, constraints in $\mathbf{C}$, and consequently also in $\mathbf{C}^*$, are sets of *unrestricted* (pointed) Kripke structures. Although $\mathbf{C}$ is not an S4 constraint system, from the above theorem, its induced scs $\mathbf{C}^*$ is. Furthermore, just like $\mathbf{C}^{\mathtt{rt}}$,

we can give in $\mathbf{C}^*$ a sound and complete compositional interpretation of formulae in S4.

We now define the compositional interpretation in our constraint system $\mathbf{C}^*$ of modal formulae. Notice that $\mathbf{C}^*$ is a powerset ordered by reversed set inclusion, hence it is a frame (Remark 3.2.1). Recall our definition of the negation constraint $\sim c$ for frames (Definition 3.2.1).

Definition 5.3.2. *Let ϕ be a modal formula from the modal language $\mathcal{L}_n(\Phi)$. The interpretation of ϕ in $\boldsymbol{C}^*$ is inductively defined as follows:*

$$\begin{aligned} \boldsymbol{C}^*[\![p]\!] &= \{(M,s) \in \Delta |\ \pi_M(s)(p) = 1\ \} \\ \boldsymbol{C}^*[\![\phi \wedge \psi]\!] &= \boldsymbol{C}^*[\![\phi]\!] \sqcup \boldsymbol{C}^*[\![\psi]\!] \\ \boldsymbol{C}^*[\![\neg\phi]\!] &= \sim \boldsymbol{C}^*[\![\phi]\!] \\ \boldsymbol{C}^*[\![\Box_i\phi]\!] &= [\![\ \boldsymbol{C}^*[\![\phi]\!]\ ]\!]_i \end{aligned}$$

where Δ is the set of all pointed Kripke structures (M,s) such that $M \in \mathcal{M}_n(\Phi)$.

Notice that $\Box_i\phi$ is interpreted in terms of the global operation. Since $\mathbf{C}^*$ is a power-set ordered by reversed inclusion the lub is given by set intersection. Thus, from Equation 5.3.1

$$\mathbf{C}^*[\![\Box_i\phi]\!] = [\![\ \mathbf{C}^*[\![\phi]\!]\ ]\!]_i = \bigsqcup_{j=0}^{\omega} [\mathbf{C}^*[\![\phi]\!]]_i^j = \bigcap_{j=0}^{\omega} [\mathbf{C}^*[\![\phi]\!]]_i^j \tag{5.3.3}$$

In particular, notice that from Theorem 5.3.1 and Axiom EP.2 it follows that $\mathbf{C}^*[\![\Box_i\phi]\!] = \mathbf{C}^*[\![\Box_i(\Box_i\phi)]\!]$ as expected for an S4-knowledge modality; i.e., if agent i knows ϕ he knows that he knows it.

We conclude this chapter with the following theorem stating the correctness wrt validity of the interpretation of knowledge as as global operator.

Theorem 5.3.2. *$\boldsymbol{C}^*[\![\phi]\!] = true$ if and only if ϕ is S4-valid.*

Proof. Let Δ be the set of all pointed Kripke structures (M,s) such that $M \in \mathcal{M}$. Similarly, let $\Delta^{\mathtt{rt}}$ be the set of all pointed Kripke structures (M,s) such that $M \in \mathcal{M}^{\mathtt{rt}}$. Given $M \in \mathcal{M}$, we use $M^* \in \mathcal{M}^{\mathtt{rt}}$ to denote the Kripke structure that results from M by replacing its accessibility relations with their corresponding transitive and reflexive closure.

From Definitions 4.2.1 and 5.3.1 we conclude that $\Delta = \mathit{true}$ in $\mathbf{C}^*$ and $\Delta^{\mathtt{rt}} = \mathit{true}$ in $\mathbf{C}^{\mathtt{rt}}$. From Definitions 4.2.1 and Proposition 5.2.2, it suffices to prove that

$$\mathbf{C}^*[\![\phi]\!] = \Delta \text{ if and only if } \mathbf{C}^{\mathtt{rt}}[\![\phi]\!] = \Delta^{\mathtt{rt}} \tag{5.3.4}$$

Property 5.3.4 is a corollary of the following two properties:

$$\text{For all } (M,s) \in \Delta^{\mathtt{rt}} : \ (M,s) \in \mathbf{C}^{\mathtt{rt}}[\![\phi]\!] \text{ iff } (M,s) \in \mathbf{C}^*[\![\phi]\!] \tag{5.3.5a}$$

$$\text{For all } (M,s) \in \Delta : \ (M,s) \in \mathbf{C}^*[\![\phi]\!] \text{ iff } (M^*,s) \in \mathbf{C}^*[\![\phi]\!] \tag{5.3.5b}$$

Proof of 5.3.5a Let $(M,s) \in \Delta^{\mathtt{rt}}$. We proceed by induction on the size of ϕ. The base case $\phi = p$ is trivial. For the inductive step here we show the most interesting case: $\phi = \Box_i\psi$ (the other cases follow directly from the induction hypothesis and the compositionality of the interpretations).

($\Rightarrow$) Assume $(M,s) \in \mathbf{C}^{\mathtt{rt}}[\![\Box_i\psi]\!]$. From Equation 5.3.3 we want to prove that $(M,s) \in \bigcap_{j=0}^{\omega}[\mathbf{C}^*[\![\psi]\!]]_i^j$. Take an arbitrary sequence $s_1, s_2, \ldots$ such that $s = s_0 \xrightarrow{i}_M s_1 \xrightarrow{i}_M s_2 \xrightarrow{i}_M \ldots$. From Definition 4.2.1 and Equation 5.3.3 it suffices to show that $(M,s_k) \in \mathbf{C}^*[\![\psi]\!]$ for $k = 0,1,\ldots$. From the assumption and Definition 4.2.1 we know that for every t such that $s \xrightarrow{i}_M t$ we have $(M,t) \in \mathbf{C}^{\mathtt{rt}}[\![\psi]\!]$. From the assumption we also know that $\xrightarrow{i}_M$ is transitive and reflexive: We thus conclude that $(M,s_k) \in \mathbf{C}^{\mathtt{rt}}[\![\psi]\!]$ for $k = 0,1,\ldots$. From the induction hypothesis, we derive $(M,s_k) \in \mathbf{C}^*[\![\psi]\!]$ for $k = 0,1,\ldots$ as wanted.

($\Leftarrow$) Assume $(M,s) \in \mathbf{C}^*[\![\phi]\!]$. From Equation 5.3.3, $(M,s) \in \bigcap_{j=0}^{\omega}[\mathbf{C}^*[\![\psi]\!]]_i^j$. Then $(M,s) \in [\mathbf{C}^*[\![\psi]\!]]_i$. From Definition 4.2.1, for every t such that $s \xrightarrow{i}_M t$ we have $(M,t) \in \mathbf{C}^*[\![\psi]\!]$. From the induction hypothesis, we can conclude that $(M,t) \in \mathbf{C}^{\mathtt{rt}}[\![\psi]\!]$ for every t such that $s \xrightarrow{i}_M t$. This shows that $(M,s) \in [\mathbf{C}^{\mathtt{rt}}[\![\psi]\!]]_i$ as wanted since $[\mathbf{C}^{\mathtt{rt}}[\![\psi]\!]]_i = \mathbf{C}^{\mathtt{rt}}[\![\phi]\!]$.

Proof of 5.3.5b Let $(M,s) \in \Delta$. We proceed by induction on the size of ϕ. The base case $\phi = p$ is trivial. For the inductive step, we show the case $\phi = \Box_i\psi$ (as in the previous proof the other cases follow directly from the induction hypothesis and the compositionality of the interpretations).

($\Rightarrow$) Assume $(M,s) \in \mathbf{C}^*[\![\phi]\!]$. Take an arbitrary sequence $s_1, s_2, \ldots$ such that $s = s_0 \xrightarrow{i}_{M^*} s_1 \xrightarrow{i}_{M^*} s_2 \xrightarrow{i}_{M^*} \ldots$. From Equation 5.3.3 it suffices to show that

$(M^*, s_k) \in \mathbf{C}^*[\![\psi]\!]$ for $k = 0, 1, \ldots$. Notice that $\xrightarrow{i}_{M^*}$ is the transitive and reflexive closure of $\xrightarrow{i}_M$, thus we have $s(\xrightarrow{i}_{M^*})^* s_k$ if and only if $s \xrightarrow{i}_{M^*} s_k$. Consequently, let us then take an arbitrary t such that $s \xrightarrow{i}_{M^*} t$: It is sufficient to show that $(M^*, t) \in \mathbf{C}^*[\![\psi]\!]$. Since $\xrightarrow{i}_{M^*}$ is the transitive and reflexive closure of $\xrightarrow{i}_M$, there must exist $s = t_0 \xrightarrow{i}_M t_1 \xrightarrow{i}_M t_2 \xrightarrow{i}_M \ldots$ such that $t_j = t$ for some $j \geq 0$. From the assumption and Equation 5.3.3 we have $(M, s) \in \bigcap_{j=0}^{\omega} [\mathbf{C}^*[\![\psi]\!]]_i^j$. Thus for any $t_1, t_2, \ldots$ such that $s = t_0 \xrightarrow{i}_M t_1 \xrightarrow{i}_M t_2 \xrightarrow{i}_M \ldots$ we have $(M, t_k) \in \mathbf{C}^*[\![\psi]\!]$ for $k = 0, 1, \ldots$. We conclude that $(M, t) \in \mathbf{C}^*[\![\psi]\!]$, and thus from the induction hypothesis we obtain $(M^*, t) \in \mathbf{C}^*[\![\psi]\!]$ as wanted.

($\Leftarrow$) Assume $(M^*, s) \in \mathbf{C}^*[\![\phi]\!]$. Take an arbitrary sequence $s_1, s_2, \ldots$ such that $s = s_0 \xrightarrow{i}_M s_1 \xrightarrow{i}_M s_2 \xrightarrow{i}_M \ldots$. From Equation 5.3.3, it suffices to show that $(M, s_k) \in \mathbf{C}^*[\![\psi]\!]$ for $k = 0, 1, \ldots$. Notice that $s \xrightarrow{i}_{M^*} s_k$ for $k = 0, 1, \ldots$ since $\xrightarrow{i}_{M^*}$ is the transitive and reflexive closure of $\xrightarrow{i}_M$. From the assumption and Equation 5.3.3 we know that for every t such that $s \xrightarrow{i}_{M^*} t$ we have $(M^*, t) \in \mathbf{C}^*[\![\psi]\!]$. We then conclude that for $k = 0, 1, \ldots$, $(M^*, s_k) \in \mathbf{C}^*[\![\psi]\!]$. We use the induction hypothesis to conclude that $(M, s_k) \in \mathbf{C}^*[\![\psi]\!]$ for $k = 0, 1, \ldots$ as wanted. □

5.4 Summary

In this chapter we created a variation of the Kripke constraint system where the space function is a closure operator that mimics S4 knowledge (i.e. the space function is extensive and idempotent). We then proceeded to encode S4 knowledge using a scs with the global information operator from Definition 2.2.3 restricted to one agent as the space function. We then proved the completeness of this constraint system with respect to S4 epistemic logic formulae. The work presented in this chapter is taken from [GHP+16].

In the next chapter we will study how concepts of time such as future and past can be properly interpreted as space and extrusion in our constraint systems.

Chapter 6

Algebraic view of LTL

In this chapter we device a mechanism to specify time properties using spatial constraint systems with extrusion. For this, we take Linear Temporal Logic (LTL) and give its modalities a semantic interpretation in terms of space and extrusion. The idea is to encode the semantic model of LTL (sequences of states) [PM92] in the form of a constraint system. We then denote future as the space function and proceed to derive an extrusion, which in turn will denote past. Kripke semantics for temporal logics have been studied before [Ryb97], thus it is theoretically possible to instead use our Kripke scs. Regardless, we shall take the direct approach of creating a new constraint system encoding the semantic model of LTL.

6.1 Linear Temporal Logic

Linear Temporal Logic (LTL) is a formalism initially conceived for specification of program properties, its main aim is to provide a logic for expressing timed behavior. A comprehensive guide for LTL can be found in [PM92], nonetheless we will introduce it in two parts accordingly to its expressivity regarding time behavior.

Future

Definition 6.1.1 (LTL Language)**.** *Given a set* Φ *of atomic propositions, the language* $\mathcal{L}^{LTL}(\Phi)$ *is the set of modal formulas constructed with the following syntax:*

$$\varphi, \psi, \ldots := p \mid \varphi \wedge \psi \mid \neg\varphi \mid \bigcirc \varphi \mid \Box \varphi$$

where $p \in \Phi$. Logical operators $\Rightarrow$, $\Leftrightarrow$ and $\vee$ are defined in the standard way.

We give intuitions on the modal operators. The formula $\bigcirc\varphi$ (read as *next* φ) means "in the next instant, φ holds" and formula $\Box\varphi$ (read as *always* φ) is interpreted as "henceforth, φ is true". We define a dual of $\Box$ as $\Diamond\varphi \stackrel{\texttt{def}}{=} \neg\Box\neg\varphi$ (read as *eventually* φ) meaning "φ will eventually be true". We proceed to describe a semantic model for LTL which is similar to that of Definition 4.1.2.

Definition 6.1.2 (LTL model)**.** *Given a set Φ of atomic propositions, a* model *M over Φ is a pair (S, π) where:*

— *$S = s_0, s_1, s_2, \ldots$ is a sequence of states,*

— *$\pi : S \to (\Phi \to \{0, 1\})$ is an interpretation for every $p \in \Phi$.*

Notation 6.1.1. *The states of model M are considered* instants *in a timeline (denoted by S_M). Similar to Notation 4.1.2, we write $\mathcal{L}^{LTL}$ when Φ is unimportant and use π_M to stand for the interpretation π from model M.*

Having defined the semantic model we are now able to formalize the semantic meaning of an $\mathcal{L}^{\text{LTL}}$ formula. Similarly to the semantics of modal logics, we say φ holds in (M, s_i) whenever $(M, s_i) \models \varphi$. Validity and satisfiability in a model M is defined in the same manner as in modal logic.

Definition 6.1.3 (LTL semantics)**.** *Consider a relation $\models$ between states of LTL models and formulas from the set $\mathcal{L}^{LTL}(\Phi)$. We define it inductively as follows:*

- $(M, s_i) \models p$ *iff* $\pi_M(s_i)(p) = 1$
- $(M, s_i) \models \varphi \wedge \psi$ *iff* $(M, s_i) \models \varphi$ *and* $(M, s_i) \models \psi$
- $(M, s_i) \models \neg\varphi$ *iff* $(M, s_i) \not\models \varphi$
- $(M, s_i) \models \bigcirc \varphi$ *iff* $(M, s_{i+1}) \models \varphi$
- $(M, s_i) \models \Box\varphi$ *iff* $(M, s_j) \models \varphi$ *for all* $j \geq i$
- $(M, s_i) \models \Diamond\varphi$ *iff* *there exists* $j \geq i$ *such that* $(M, s_j) \models \varphi$

Past

Reasoning about properties in the past is also a desired feature. In order to obtain this, we redefine the LTL language by adding past operators.

Definition 6.1.4 (LTL Language)**.** *Given a set* Φ *of atomic propositions, the language* $\mathcal{L}^{LTL}(\Phi)$ *is the set of modal formulas constructed with the following syntax:*

$$\varphi, \psi := p \mid \varphi \wedge \psi \mid \neg\varphi \mid \bigcirc\varphi \mid \Box\varphi \mid \ominus\varphi \mid \boxminus\varphi$$

where $p \in \Phi$*. Logical operators* $\Rightarrow$*,* $\Leftrightarrow$ *and* $\vee$ *are defined in the standard way.*

The intuition on the future operators (i.e. *next*, *always* and *eventually*) remain the same. The formula $\ominus\varphi$ (read as *previous* φ) means "in the last instant, φ was true". Formula $\boxminus\varphi$ is interpreted as "φ *has always been* true" and finally, we also define a dual of $\boxminus$ as $\diamondminus\varphi \stackrel{\texttt{def}}{=} \neg\boxminus\neg\varphi$ (read as *once* φ) meaning "φ was once true".

Again, we use the model defined in 6.1.2 to complement the semantics of LTL by including the past operators. The semantics of future operators remain the same.

Definition 6.1.5 (LTL-Semantics)**.** *We complement Definition 6.1.3 with the following:*

- $(M, s_i) \models \ominus\,\varphi \quad$ *iff* $\quad i > 0$ *and* $(M, s_{i-1}) \models \varphi$
- $(M, s_i) \models \boxminus\varphi \quad$ *iff* $\quad (M, s_j) \models \varphi$ *for all* $j \leq i$
- $(M, s_i) \models \diamondminus\varphi \quad$ *iff* $\quad$ *there exists* $j \leq i$ *such that* $(M, s_j) \models \varphi$

Weak Past

Because state s_0 does not have a past, $(M, s_0) \models \ominus\varphi$ is defined as *false* for all models M and all formulas φ. We can define a *weaker* version of the *previous* operator (written as $\tilde{\ominus}$).

Definition 6.1.6 (Weak past semantics)**.** *The* weak past *operator is semantically defined as follows:*

- $(M, s_i) \models \tilde{\ominus}\,\varphi \quad$ *iff* $\quad i = 0$ *or* $(M, s_{i-1}) \models \varphi$

Using Definition 6.1.6 we can obtain the next theorem:

$$\tilde{\ominus}\ \varphi \Leftrightarrow \neg \ominus \neg\varphi$$

Moreover, $(M, s_0) \models \tilde{\ominus}\varphi$ is equals to *true* for all models M and all formulas φ. This allows us to identify the first instant in time with the following definition:

$$\mathit{first} \stackrel{\texttt{def}}{=} \tilde{\ominus}\ \mathit{false}$$

6.2 Constraint systems for LTL models

The semantic model of LTL (Definition 6.1.2) uses an infinite sequence of states to represent the timeline of a program execution [PM92]. We will encode this semantic structure in a constraint system following the same methodology used in our Krikpe scs (Definition 4.2.1).

Definition 6.2.1 (Time scs)**.** *Let $\mathcal{S}(\Phi)$ be a non-empty set of LTL models over Φ. Let Δ be the set of all pairs (M, s_i) such that $M \in \mathcal{S}(\Phi)$ and s_i is a state of the sequence S_M. We define the Time scs for $\mathcal{S}(\Phi)$ as the following single agent scs:*

$$\mathbf{T}(\mathcal{S}(\Phi)) = (\mathit{Con}, \sqsubseteq, [\cdot])$$

where $\mathit{Con} = \mathcal{P}(\Delta)$, $c_1 \sqsubseteq c_2$ iff $c_2 \subseteq c_1$ and for every $c \in \mathit{Con}$

$$[c] \stackrel{\texttt{def}}{=} \{(M, s_i) \in \Delta \mid (M, s_{i+1}) \in c\}.$$

Remark 6.2.1. *Just as the Kripke scs, $\mathbf{T}(\mathcal{S}(\Phi))$ is a complete algebraic lattice given by a powerset ordered by $\supseteq$ where $\sqcup$ is the set intersection, the top element false is $\emptyset$ and bottom element true is Δ. Considering they are of the form $(\mathcal{P}(U), \supseteq)$ where meets are unions and joins intersections, we can also conclude that $\mathbf{T}(\mathcal{S}(\Phi))$ is a frame and that the operators $\rightarrow$ and $\sim$ coincide with the constructions $\Rightarrow$ and $\neg$ defined in Remark 2.2.1.*

We nonetheless need to account for it being a spatial cs.

Proposition 6.2.1. *Let $\mathbf{T}(\mathcal{S}(\Phi)) = (\mathit{Con}, \sqsubseteq, [\cdot])$ be a Time scs. The function $[\cdot]$ is a space function.*

Proof. We show that $[\cdot]$ fulfills axioms S.1 and S.2:

For S.1:

$[\mathit{true}]$ by definition is $[\Delta] = \{(M, s_i) \in \Delta \mid (M, s_{i+1}) \in \Delta\}$. Nonetheless every $(M, s_i) \in \Delta$ consequently $[\Delta] = \Delta$ and $[\mathit{true}] = \mathit{true}$.

For S.2:

$$\begin{aligned}
[c \sqcup d] = &\quad \{(M, s_i) \in \Delta \mid (M, s_{i+1}) \in c \sqcup d\} \\
= &\quad \{(M, s_i) \in \Delta \mid (M, s_{i+1}) \in c \cap d\} \\
= &\quad \{(M, s_i) \in \Delta \mid (M, s_{i+1}) \in c\} \cap \{(M, s) \in \Delta \mid (M, s_{i+1}) \in d\} \\
= &\quad [c] \cap [d] \\
= &\quad [c] \sqcup [d]
\end{aligned}$$

□

Because surjectivity and completeness (as in Definition 2.1.14) are central properties of our extrusion operator we also verify their validity in Time scs.

Proposition 6.2.2. *The space function of* $\mathbf{T}(\mathcal{S}(\Phi))$ *is surjective and both join-complete and meet-complete.*

Proof.

The space function is surjective:

Let $d \in \mathit{Con}$ and c be the set $\{(M, s_{i+1}) \in \Delta \mid (M, s_i) \in d\}$. Because $c \in \mathit{Con}$ and $[c] = d$ we conclude that $[\cdot]$ is surjective. Alternatively, you can see the sequence $s_0, s_1, \ldots$ as a *left-total* and *left-unique* relation, therefore Proposition 4.3.2 applies.

The space function is join-complete:

We need to prove $[\bigcap S] = \bigcap [S]$. By definition $(M, s_j) \in \bigcap S \Leftrightarrow \forall_{c \in S}\ (M, s_j) \in c$. We then conclude

$$\{(M, s_i) \in \Delta \mid (M, s_{i+1}) \in \bigcap S\} = \bigcap_{c \in S} \{(M, s_i) \in \Delta \mid (M, s_{i+1}) \in c\}$$

as wanted.

The space function is meet-complete:

We need to prove $[\bigcup S] = \bigcup[S]$. By definition $(M, s_j) \in \bigcup S \Leftrightarrow \exists_{c \in S}\ (M, s_j) \in c$. Considering there always exists a unique successor to any state s_i (i.e. s_{i+1}), we can then conclude

$$\{(M, s_i) \in \Delta \mid (M, s_{i+1}) \in \bigcup S\} = \bigcup_{c \in S} \{(M, s_i) \in \Delta \mid (M, s_{i+1}) \in c\}$$

as wanted. □

Having proved that the space function of $\mathbf{T}(\mathcal{S}(\Phi))$ is surjective as well as meet and join-complete, we proceed to apply Theorems 3.4.3 and 3.4.1. We define the next extrusion functions:

$$\uparrow : c \mapsto \bigsqcup [c]^{-1} \qquad\qquad \tilde{\uparrow} : c \mapsto \bigsqcap [c]^{-1}$$

As we also did in Section 4.3, we obtain the intensional definitions of the above extrusion functions.

Lemma 6.2.1. *Consider the spatial cs* $\mathbf{T}(\mathcal{S}(\Phi))$ *and let* $\uparrow(c) = \bigsqcup [c]^{-1}$ *and* $\tilde{\uparrow}(c) = \bigsqcap [c]^{-1}$*, we then have:*

$$\uparrow(c) = \{(M, s_i) \in \Delta \mid i > 0 \text{ and } (M, s_{i-1}) \in c\}$$
$$\tilde{\uparrow}(c) = \{(M, s_i) \in \Delta \mid i = 0 \text{ or } (M, s_{i-1}) \in c\}$$

Proof.

We need to prove that $(M, s_i) \in \uparrow(c)$ iff $(M, s_i) \in \bigcap [c]^{-1}$.

— We first prove that if $(M, s_i) \in \uparrow(c)$ then $(M, s_i) \in \bigcap [c]^{-1}$, for suppose not. This means $(M, s_{i-1}) \in c$, $i > 0$ and that $\exists d \in [c]^{-1}$ s.t. $(M, s_i) \notin d$. Because $[d] = c$ we have that $c = \{(M, s_j) \in \Delta \mid (M, s_{j+1}) \in d\}$, and together with the fact that $(M, s_{i-1}) \in c$ we arrive to the contradiction $(M, s_i) \in d$.

— We proceed to prove that if $(M, s_i) \in \bigcap [c]^{-1}$ then $(M, s_i) \in \uparrow(c)$. Suppose $(M, s_i) \in \bigcap [c]^{-1}$ then $\forall_{d \in Con}$ if $[d] = c$ then $(M, s_i) \in d$. Subsequently, we only need to prove that $[\uparrow(c)] = c$ which is true as $[\uparrow(c)] = \{(M, s_j) \in \Delta \mid (M, s_{j+1}) \in \{(M, s_k) \in \Delta \mid k > 0 \text{ and } (M, s_{k-1}) \in c\}\} = \{(M, s_j) \in \Delta \mid (M, s_j) \in c\} = c$.

Additionally, we also prove that $(M, s_i) \in \tilde{\uparrow}(c)$ iff $(M, s_i) \in \bigcup[c]^{-1}$ for every $c \in Con$.

— We begin proving that if $(M, s_i) \in \tilde{\uparrow}(c)$ then $(M, s_i) \in \bigcup[c]^{-1}$. Suppose $(M, s_i) \in \tilde{\uparrow}(c)$ thus if $[\tilde{\uparrow}(c)] = c$ then $\tilde{\uparrow}(c) \in [c]^{-1}$ and subsequently $(M, s_i) \in \bigcup[c]^{-1}$. Furthermore $[\tilde{\uparrow}(c)] = \{(M, s_j) \in \Delta \mid (M, s_{j+1}) \in \{(M, s_k) \in \Delta \mid k = 0 \text{ or } (M, s_{k-1}) \in c\}\} = \{(M, s_j) \in \Delta \mid (M, s_j) \in c\} = c$ as wanted.

— And finally we prove that if $(M, s_i) \in \bigcup[c]^{-1}$ then $(M, s_i) \in \tilde{\uparrow}(c)$, for suppose not. We then have that $\exists d \in [c]^{-1} s.t. (M, s_i) \in d$, also we have that $i > 0$ and $(M, s_{i-1}) \notin c$. Because $[d] = c = \{(M, s_j) \in \Delta \mid (M, s_{j+1}) \in d\}$ and the fact that $(M, s_{i-1}) \notin c$ we arrive to the contradiction $(M, s_i) \notin d$.

□

After defining the extrusion we proceed to extend the Time scs $\mathbf{T}(\mathcal{S}(\Phi))$ to an spatial constraint system with extrusion as it is shown in the next definition:

Definition 6.2.2. *We define the Time scs-e* $\mathbf{T}^{\uparrow}(\mathcal{S}(\Phi))$ *as the Time scs* $\mathbf{T}(\mathcal{S}(\Phi))$ *extended with the following extrusion function:*

$$\uparrow(c) = \{(M, s_i) \in \Delta \mid i > 0 \text{ and } (M, s_{i-1}) \in c\}$$

6.3 Constraint system semantics for LTL

We shall now give semantics to an LTL formula using the scs-e $\mathbf{T}^{\uparrow}(\mathcal{S}(\Phi))$ of Definition 6.2.2.

Definition 6.3.1. *Let* $\mathbf{T}^{\uparrow}(\mathcal{S}(\Phi)) = (Con, \sqsubseteq, [\cdot], \uparrow(\cdot))$ *be the scs-e in Definition 6.2.2. Given a formula* φ *from the language* $\mathcal{L}^{LTL}(\Phi)$*, its denotation* $\mathbf{T}^{\uparrow}[\![\varphi]\!]$ *is*

inductively defined as follows:

$$\begin{aligned}
\mathbf{T}^{\uparrow}[\![p]\!] &= \{(M, s_i) \in \Delta \mid \pi_M(s_i)(p) = 1\} \\
\mathbf{T}^{\uparrow}[\![\varphi \wedge \psi]\!] &= \mathbf{T}^{\uparrow}[\![\varphi]\!] \sqcup \mathbf{K}^{\uparrow}[\![\psi]\!] \\
\mathbf{T}^{\uparrow}[\![\neg\varphi]\!] &= \sim \mathbf{T}^{\uparrow}[\![\varphi]\!] \\
\mathbf{T}^{\uparrow}[\![\bigcirc\varphi]\!] &= [\ \mathbf{T}^{\uparrow}[\![\varphi]\!]\ ] \\
\mathbf{T}^{\uparrow}[\![\Box\varphi]\!] &= [\![\ \mathbf{T}^{\uparrow}[\![\varphi]\!]\ ]\!] \\
\mathbf{T}^{\uparrow}[\![\ominus\varphi]\!] &= \uparrow \mathbf{T}^{\uparrow}[\![\varphi]\!] \\
\mathbf{T}^{\uparrow}[\![\boxminus\varphi]\!] &= \Uparrow \mathbf{T}^{\uparrow}[\![\varphi]\!]
\end{aligned}$$

where Δ is the set of all pairs (M, s_i) such that $M \in \mathcal{S}(\Phi)$. A formula φ of $\mathcal{L}^{LTL}(\Phi)$ is valid iff $\mathbf{T}^{\uparrow}[\![\varphi]\!] = true$.

Remark 6.3.1. *While it is easy to see that, semantically speaking, the LTL modality $\ominus$ is the right inverse of the $\bigcirc$ modality, we cannot say the same about the $\boxminus$ and $\Box$ LTL modalities (see the equations after Definition 3.2.2).*

Given the above mentioned semantics, and using the axioms and properties of spatial constraint systems with extrusion, one is now able to verify the validity of axioms from the $\mathcal{L}^{\mathrm{LTL}}$ language.

Lemma 6.3.1. *The LTL formula $\tilde{\ominus}\varphi \Leftrightarrow \neg \ominus \neg\varphi$ is valid.*

Proof. We split the formula into $\tilde{\ominus}\varphi \Rightarrow \neg \ominus \neg\varphi$ and $\neg \ominus \neg\varphi \Rightarrow \tilde{\ominus}\varphi$. According to the concept of validity in Definition 6.3.1 we need to prove that $\mathbf{T}^{\uparrow}[\![\tilde{\ominus}\varphi \Rightarrow \neg \ominus \neg\varphi]\!] = true$ and $\mathbf{T}^{\uparrow}[\![\neg \ominus \neg\varphi \Rightarrow \tilde{\ominus}\varphi]\!] = true$ which semantically is the same as proving $\tilde{\uparrow}\ \mathbf{T}^{\uparrow}[\![\varphi]\!] \rightarrow \sim\uparrow\sim \mathbf{T}^{\uparrow}[\![\varphi]\!] = true$ and $\sim\uparrow\sim \mathbf{T}^{\uparrow}[\![\varphi]\!] \rightarrow\ \tilde{\uparrow}\ \mathbf{T}^{\uparrow}[\![\varphi]\!] = true$. By using Lemma 3.2.2 it is sufficient to show that $\tilde{\uparrow}\ \mathbf{T}^{\uparrow}[\![\varphi]\!] \sqsupseteq \sim\uparrow\sim \mathbf{T}^{\uparrow}[\![\varphi]\!]$ and $\sim\uparrow\sim \mathbf{T}^{\uparrow}[\![\varphi]\!] \sqsupseteq \tilde{\uparrow}\ \mathbf{T}^{\uparrow}[\![\varphi]\!]$ that subsequently is the same as proving $\tilde{\uparrow}\ \mathbf{T}^{\uparrow}[\![\varphi]\!] = \sim\uparrow\sim \mathbf{T}^{\uparrow}[\![\varphi]\!]$. To prove this final form, we use the intensional definitions in Lemma 6.2.1 and replace $\sim$ with set complement as dictated in Remark 2.2.1 as follows: $\sim\uparrow\sim \mathbf{T}^{\uparrow}[\![\varphi]\!] = \Delta\backslash\{(M, s_i) \mid i > 0 \text{ and } (M, s_{i-1}) \in \Delta\backslash\mathbf{T}^{\uparrow}[\![\varphi]\!]\} = \Delta\backslash\{(M, s_i) \mid i > 0 \text{ and } (M, s_{i-1}) \notin \mathbf{T}^{\uparrow}[\![\varphi]\!]\} = \{(M, s_i) \mid i = 0 \text{ or } (M, s_{i-1}) \in \mathbf{T}^{\uparrow}[\![\varphi]\!]\} = \tilde{\uparrow}\ \mathbf{T}^{\uparrow}[\![\varphi]\!]$. □

As mentioned before, the weak past operator $\tilde{\ominus}$ is the dual of the past operator $\ominus$. However, for the sake of completeness we also give cs semantics to the weak past operator.

Definition 6.3.2. *Let $\mathbf{T}^{\uparrow}(\mathcal{S}(\Phi))$ be the scs-e in Definition 6.2.2. We augment the denotation $\mathbf{T}^{\uparrow}[\![\varphi]\!]$ of formula φ from $\mathcal{L}^{LTL}(\Phi)$ to include the operator $\tilde{\ominus}$ as follows:*

$$\mathbf{T}^{\uparrow}[\![\tilde{\ominus}\varphi]\!] = \tilde{\uparrow}\ \mathbf{T}^{\uparrow}[\![\varphi]\!]$$

Furthermore, using the extensional definitions in 6.2.1 and the semantics of Definition 6.3.1 we can verify the claim of the formula $\tilde{\ominus}\mathit{false}$ capturing the initial states of the LTL models.

Proposition 6.3.1. *if $(M, s_i) \in \ \mathbf{T}^{\uparrow}[\![\tilde{\ominus}\mathit{false}]\!]$ then $i = 0$.*

Proof. $\mathbf{T}^{\uparrow}[\![\tilde{\ominus}\mathit{false}]\!] = \tilde{\uparrow}\ \mathbf{T}^{\uparrow}[\![\mathit{false}]\!] = \tilde{\uparrow}(\emptyset) = \{(M, s_i) \in \Delta \mid i = 0 \text{ or } (M, s_i) \in \emptyset\} = \{(M, s_i) \in \Delta \mid i = 0\}$. □

6.4 Summary

After introducing Linear Temporal Logic and following the same methodology of the last two chapters, we defined Time constraint systems as our main structures to encode time properties. We derived two extrusion operators using Theorems 3.4.1 and 3.4.3, which were then later shown to have the same intentional description as those of the normal and weak past operators of LTL. Moreover, we gave a semantic characterization of LTL formulae along with the definition of validity. Using this semantic interpretation we showed the relation between the normal and weak past operators and proved the ability of a specific LTL formula to capture the initial instant of time. Although the results here are complete, their publication is still a work in progress and will be done accordingly.

Chapter 7

Conclusions and Related Work

We finalize this document with concluding remarks and expand further on the problem of extrusion. We give some related approaches mainly at the conceptual level of the inverse of space and potential applicability of our results. We also discuss possible future research to enhance the epistemic expressivity of constraint systems and their relation to multi-agent interactive systems.

In this thesis we studied spatial constraint system (scs); complete lattices equipped with self-maps accounting for the concept of agent spaces. These maps are called *space functions*. We then proceeded to define spatial constraint systems with extrusion (scse); an extension of scs with a second set of self-maps called *extrusion functions*. We regard extrusion functions as the right inverse of space functions.

We formalized scse by building upon notions and concepts from order (domain) theory, epistemic (doxastic) theories, and the algebraic treatment of logic in [Vic96, DPPB97]. Alternatively, we adapted the definition of implication in Heyting Algebras [Vic96] to our constraint systems in order to have operators for negation and implication of elements. With these new concepts, we were able to model situations with concurrent and epistemic behaviors such as process mobility, communicating agents and hoaxes.

Furthermore, we addressed the problem of constructing an extrusion function given a space function. This is an important aspect because building up the constraint system anew might not be an option; the user may wish to extend

a previous scs with some fixed space functions. Supposing the axiom of choice, the sufficient and necessary condition for the existence of an extrusion function is surjectivity. Checking surjectivity on the space function might be expensive (depending on the number of elements of the cs), therefore we gave a necessary condition significantly easier to check (i.e. space consistency).

We also stated sufficient conditions for explicit constructions of an extrusion function given a space function. One of these explicit constructions allows the extrusion function to be distributive and strict (properties that space functions fulfill by definition). Finally, we also studied properties of space and extrusion such as space consistency, completeness, automorphisms and Galois connections.

As a first application of our results, we devised an epistemic logic of belief and utterance where the modality of utterance is the right inverse of the belief modality. We also defined common epistemic behaviors such as opinions and hoaxes, and then proceeded to characterize the logic by giving semantics for each operator and modality. The semantics were defined in terms of a Kripke scse and the two modalities were also demonstrated to form a Galois connection.

We also showed that Kripke spatial constraint system can represent S4 knowledge by using a global space construction. It was then proved that this representation fulfills the S4 axioms of knowledge -i.e., extensiveness and positive introspection. This encoding was also demonstrated to be sound and complete with respect to formulae of the S4 axiom system.

Finally, we created Time constraint systems and used them to encode formulas of LTL. With this, we opened the possibility of expressing time behavior using algebraic expressions from scs with extrusion. A particular feature was the semantic definition of a normal and weak versions of the past operator that were shown to correspond to the explicit constructions of extrusion in the first half of this thesis.

All in all, we have argued that belief, knowledge and utterances correspond, respectively, to the spatial concepts of local space, global space, and extrusion. Alternatively, the concepts of future and past also correspond to space and extrusion, where different versions of the past may be definable.

The Table 7.1 in the Appendix section gives a summary of the correspondence between lattice operators of spatial constraint systems, their instantiation in the Kripke constraint system, and their epistemic and time counterparts.

7.1 Related and Future Work

Our scse's can be used as constraint systems for concurrent constraint programming (ccp) calculi [SRP91]. This way processes in these calculi would be able to express spatial mobility and epistemic/social behaviors. The issue of extending ccp calculi to provide for mobility and distributed information has been previously addressed in [DRV98, Rét98, GP00, BKJG02]. In [Rét98, DRV98] processes can send constraints using communication channels much like in the π-calculus. Moreover, [GP00] extends ccp with primitives for process mobility within a hierarchically organized network described as a tree. Additionally, the authors of [BKJG02] create an extension to ccp where agents maintain a local store and communicate through a global store. In [OV08] temporal ccp process can transmit variables using existential and universal quantification. More recently, in [ONP13] the authors added the notion of link mobility to spatial and linear ccp using a proof-theoretical approach. Furthermore, distributed versions of ccp in a network of nodes are studied in [BW05]. Our approach differs from these works in both conception, technical development, and applications. We view extrusion as an inverse of space and develop this concept using lattice theory and domain theory. None of the previously-mentioned works is concerned with applications for reasoning about epistemic behavior.

Another work that has influenced the design of scs's with extrusion is the ambient calculus [CG98]. Ambient provides for the specification of processes that can move in and out within their spatial hierarchy. It does not, however, address posting and querying epistemic information within a spatial distribution of processes. Our notion of extrusion is reminiscent of Ambient's notion of *subjective mobility*. In future work we plan to investigate a domain-theoretical approach to Ambient concepts such as *acid operations*. Intuitively, an acid operation can cause space to be disolved, and thus we may be able to characterize it as a function ac_i that "undoes" space, i.e., $ac_i \circ [\cdot]_i = id$.

The process calculi in [BJPV09, BM07, FRS01] provide for the use of assertions within π-like processes. They are not concerned with spatial distribution of information and knowledge. These frameworks are very generic and offer several reasoning techniques. Therefore, it would be interesting to see how the ideas here

developed can be adapted to them.

Inverse modalities have been used in temporal, epistemic and Hennessy-Milner logic. For example, in [RS97] the logical properties and consequences of introducing inverse modalities in a generic modal logic are thoroughly explained. Also in [PU11] the authors put forward an extension of Hennessy-Milner logic with a reverse modality for expressing concurrent behavior. In future work we plan to develop an algebraic presentation of scse's by building upon the axiomatization of logics with reverse modalities given in [RS97].

Epistemic logics have been widely applied to distributed systems; [FHMV95] gives a good summary of the subject. This work is all aimed at analyzing distributed protocols using epistemic logic as a reasoning tool. The work has been very influential in setting previous stages for the present work but it is not closely connected to the present proposal to put epistemic concepts into constraint systems and thus algebraic structures. Concerning time concepts, Kripke semantics for temporal logics have been studied before [Ryb97]. Nonetheless, this thesis gives a direct algebraic characterization of the semantic models of LTL used in [PM92].

Social phenomena such as lies, utterance, opinions have been recently studied in epistemic (doxastic) logic [VDVESW12, VD14, SCH10]. We follow [VDVESW12] and regard lies as utterances by an agent that are inconsistent with their beliefs. Apart from our domain-theoretical treatment of these epistemic concepts, a difference with [VDVESW12, VD14, SCH10] is that we developed utterance as an inverse modality (upper adjoint) of belief. As future work we would like to investigate how the dynamic-logic approach in [VDVESW12, SCH10, VD14] of the above-mentioned epistemic phenomena can be incorporated in our constraint systems.

The question of whether knowledge is definable in terms of belief has played an important role in epistemology. In [HSS09] the authors studied this question in the framework of epistemic and doxastic logics. They proved that epistemic logic S5 cannot/can be explicitly/implicitly defined in terms of the belief logic KD45. Here we defined belief and knowledge in terms of space. The notion of knowledge we considered corresponds to the epistemic logic S4 while the notion of belief is KD [FHMV95]. We plan to address the question of whether S5 knowledge and KD45 belief can be defined in our spatial constraint systems.

An approach closely related to ours is the spatial logics for concurrency from [CC03, CC02]. In this work they also take spatial location as the fundamental concept and develop modalities that reflect locality. Rather than using modal logic, they use the name quantifier that has been actively studied in the theory of freshness of names in programming languages. Their language is better adapted to the calculi for mobility where names play a fundamental role and the concept of freshness of a name is exploited to control the flow of information. In our framework, the epistemic concepts are manifested directly in the syntax and the semantics. It would be interesting to see how a name quantified scs would look and to study the relationship with the framework in [CC03, CC02].

Concurrent systems may allow user/agents to join the network during computation. Therefore the number of agents can be unbounded. Other potential research direction for our work is therefore to generalize spatial constraint system (and thus scs with extrusion) by allowing an unbounded number of agents.

Quantitative reasoning is also another direction where constraint systems can increase their epistemic expressivity. Currently, there is a plethora of approaches to make probabilistic extensions to modal and epistemic logic [FH94]. To the best of our knowledge no algebraic treatment of them exists. An attractive future endeavor would be an extension of our algebraic models that will allow us to add weights to agents beliefs and informations within the hierarchy of the system. The approach would be to generalize the notion of belief as a form of probabilistic measure on epistemic statements to express meaningful and ambiguous behavior commonly found in social scenarios. For instance, suppose that c represents a statement, the perception of which may change according to space, time or the individual –e.g., an inherently subjective statement about politics, religion, or sports. We may have, for example, that agent i initially believes that c holds with some probability $p < 1$. After interacting with other agents or moving across the hierarchy, agent i may end up convinced that c is true (or that it is false). We would also be able to express that agent i believes c more than agent j does and other similar statements related to social interaction. This quantitative modeling provides a more faithful representation of virtual social behavior in distributed networks, where information certainty changes parallel to its dissemination.

After introducing quantitative reasoning to our structures, a legitimate step

would be to extend and/or characterize existing epistemic logics (such as the one in [BCRS15]) that are already able to express social behavior similar to the one mentioned above. All this whilst verifying a through correspondence to the concepts of validity and satisfiability from the logic. This in turn will allows us to reason about epistemic situations and verify their validity from an algebraic point of view, as well as see the properties and interaction of the epistemic operators. For example we could verify if the social phenomena 'group polarization' is present in a certain utterance of opinions in a multi-agent environment. Group polarization refers to the tendency of groups to hold more extreme beliefs than the initial individual beliefs of its members.

7.2 Correspondence between operators

Below is a table summarizing the correspondence between the lattice operators and those of the logical languages we have characterized in this thesis. The correspondence, although developed formally here, is meant to better reflect the direct relation of the concepts of space and extrusion to more particular interpretations.

Spatial Constraint System Lattice operators	***Kripke Constraint System*** Operators on set of pointed KS's	***Epistemic Logic*** Epistemic Operators	***Linear Time Logic*** Time Operators
false (top)	$\emptyset$	F (constant false)	F (constant false)
$\sqcup$ (join)	$\cap$	$\wedge$	$\wedge$
$\sqcup$ (meet)	$\cup$	$\vee$	$\vee$
$\sim$ (pseudo complement)	complement	$\neg$	$\neg$
$[\cdot]_i$ (local space)	$i(\cdot)$	$B_i(\cdot)$ (belief)	$\bigcirc$ (next)
$\uparrow_i(\cdot)$ (extrusion)	$i^{-1}(\cdot)$	$U_i(\cdot)$ (utterance)	$\ominus$ (previous)
$[\![\cdot]\!]_i$ (global space)	$\bigcap_{k=0}^{\omega} i^k(\cdot)$	$K_i(\cdot)$ (knowledge)	$\Box$ (henceforth)

Table 7.1 – Recall that $i(X) \stackrel{\texttt{def}}{=} \{(M,s) \mid \textit{if } s \xrightarrow{i}_M t \textit{ then } (M,t) \in X\}$, $i^{-1}(X) \stackrel{\texttt{def}}{=} \{(M,t) \mid \exists s \textit{ s.t. } s \xrightarrow{i}_M t \textit{ and } (M,s) \in X\}$, $i^{k+1}(X) \stackrel{\texttt{def}}{=} i(i^k(X))$ and $i^0(X) \stackrel{\texttt{def}}{=} X$.

Bibliography

[ABP+11] Andrés Aristizábal, Filippo Bonchi, Catuscia Palamidessi, Luis Pino, and D. Valencia, Frank. Deriving labels and bisimilarity for concurrent constraint programming. In *Proceedings of the 14th International Conference on Foundations of Software Science and Computation Structures, FOSSACS 2011*, LNCS, pages 138–152. Springer, 2011.

[AJ94] Samson Abramsky and Achim Jung. Domain theory. *Handbook of logic in computer science*, pages 1–77, 1994.

[BCRS15] Alexandru Baltag, Zoé Christoff, Rasmus K Rendsvig, and Sonja Smets. Dynamic epistemic logics of diffusion and prediction in social networks. *Draftpaper, April*, 2015.

[BDPP95] Frank S. Boer, Alessandra Di Pierro, and Catuscia Palamidessi. Nondeterminism and infinite computations in constraint programming. *Theoretical Computer Science*, pages 37–78, 1995.

[BDRV02] Patrick Blackburn, Maarten De Rijke, and Yde Venema. *Modal Logic*. Cambridge University Press, 1st edition, 2002.

[BJPV09] Jesper Bengtson, Magnus Johansson, Joachim Parrow, and Björn Victor. Psi-calculi: Mobile processes, nominal data, and logic. In *Proceedings of the 24th Annual IEEE Symposium on Logic in Computer Science, LICS 2009*, pages 39–48, 2009.

[BKJG02] Lubos Brim, Mojmir Kretinsky, Jean-Marie Jacquet, and David Gilbert. Modelling multi-agent systems as synchronous concurrent constraint processes. *Computing and informatics*, 21:565–590, 2002.

[Bly06] Thomas Scott Blyth. *Lattices and ordered algebraic structures*. Springer Science & Business Media, 2006.

[BM07] Maria Grazia Buscemi and Ugo Montanari. Cc-pi: A constraint-based language for specifying service level agreements. In *Proceedings of the 16th European Symposium on Programming Languages and Systems, ESOP 2007*, pages 18–32, 2007.

[BW05] Luca Bortolussi and Herbert Wiklicky. A distributed and probabilistic concurrent constraint programming language. In *Logic Programming*, pages 143–158. Springer, 2005.

[CC02] Luís Caires and Luca Cardelli. A spatial logic for concurrency (part ii). In *Proceedings of the 13th International Conference of Concurrency Theory, CONCUR 2002*, pages 209–225, 2002.

[CC03] Luís Caires and Luca Cardelli. A spatial logic for concurrency (part i). *Information and Computation*, pages 194–235, 2003.

[CG98] Luca Cardelli and Andrew D. Gordon. Mobile ambients. In *Proceedings of the First International Conference on Foundations of Software Science and Computation Structure, FoSSaCS'98*, pages 140–155, 1998.

[DP02] Brian A Davey and Hilary A Priestley. *Introduction to lattices and order*. Cambridge university press, 2nd edition, 2002.

[DPPB97] Alessandra Di Pierro, Catuscia Palamidessi, and Frank S. Boer. An algebraic perspective of constraint logic programming. *Journal of Logic and Computation*, pages 1–38, 1997.

[DRV98] Juan Francisco Díaz, Camilo Rueda, and Frank D. Valencia. Pi+-calculus: A calculus for concurrent processes with constraints. *December 1998 Special Issue of Best Papers presented at CLEI'97*, 1998.

[FH94] Ronald Fagin and Joseph Y Halpern. Reasoning about knowledge and probability. *Journal of the ACM (JACM)*, 41(2):340–367, 1994.

[FHMV95] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y Vardi. *Reasoning about knowledge*. MIT press Cambridge, 4th edition, 1995.

[FRS01] François Fages, Paul Ruet, and Sylvain Soliman. Linear concurrent constraint programming: Operational and phase semantics. *Information and Computation*, pages 14–41, 2001.

[GHK+03] Gerhard Gierz, Karl Heinrich Hofmann, Klaus Keimel, Jimmie D. Lawson, Michael Mislove, and Dana S. Scott. *Continuous lattices and domains*. Cambridge University Press, 1st edition, 2003.

[GHP+16] Michell Guzman, Stefan Haar, Salim Perchy, Camilo Rueda, and Frank D. Valencia. Belief, knowledge, lies and other utterances in an algebra for space and extrusion. *Accepted to the Journal of Logical and Algebraic Methods in Programming, JLAMP*, 86, 2016.

[Goo10] Kenneth R. Goodearl. *Partially ordered abelian groups with interpolation*. American Mathematical Society, 1st edition, 2010.

[GP00] David Gilbert and Catuscia Palamidessi. Concurrent constraint programming with process mobility. In *Computational Logic—CL 2000*, pages 463–477. Springer, 2000.

[GPRV16] Michell Guzman, Salim Perchy, Camilo Rueda, and Frank D. Valencia. Deriving inverse operators for modal logics. In *Accepted to the 13th International Colloquium on Theoretical Aspects of Computing, ICTAC 2016*, 2016.

[Hin62] Jaakko Hintikka. *Knowledge and belief*. Cornell Univeristy Press, 1962.

[HPRV15] Stefan Haar, Salim Perchy, Camilo Rueda, and Frank D. Valencia. An algebraic view of space/belief and extrusion/utterance for concurrency/epistemic logic. In *Proceedings of the 17th ACM SIGPLAN International Symposium on Principles and Practice of Declarative Programming, PPDP 2015*, pages 161–172, 2015.

[HSS09] Joseph Y Halpern, Dov Samet, and Ella Segev. Defining knowledge in terms of belief: The modal logic perspective. *The Review of Symbolic Logic*, pages 469–487, 2009.

[KPPV12] Sophia Knight, Catuscia Palamidessi, Prakash Panangaden, and Frank D. Valencia. Spatial and epistemic modalities in constraint-based process calculi. In *Proceedings of the 23rd International Conference on Concurrency Theory, CONCUR 2012*, pages 317–332. Springer, 2012.

[Kri63] Saul A. Kripke. Semantical analysis of modal logic i normal modal propositional calculi. *Mathematical Logic Quarterly*, pages 67–96, 1963.

[MPSS95] Nax P. Mendler, Prakash Panangaden, Philip J Scott, and RAG Seely. A logical view of concurrent constraint programming. *Nordic Journal of Computing*, pages 181–220, 1995.

[MT44] John Charles Chenoweth McKinsey and Alfred Tarski. The algebra of topology. *Annals of mathematics*, pages 141–191, 1944.

[ONP13] Carlos Olarte, Vivek Nigam, and Elaine Pimentel. Dynamic spaces in concurrent constraint programming. In *Proceedings of the 8th Workshop on Logical and Semantic Frameworks, LSFA 2013*, pages 103–121. Elsevier, 2013.

[OV08] Carlos Olarte and Frank D. Valencia. The expressivity of universal timed ccp: undecidability of monadic fltl and closure operators for security. In *Proceedings of the 10th ACM SIGPLAN International Conference on Principles and Practice of Declarative Programming, PPDP 2008*, pages 8–19, 2008.

[Per15] Salim Perchy. Can we finally agree on the 8th art? a technological approach. *Le Magazine des Élèves de L'École Polytechnique, X-Passion*, 71, 2015.

[PM92] Amir Pnueli and Zohar Manna. *The temporal logic of reactive and concurrent systems*. Springer, 1992.

[PSSS93] Prakash Panangaden, Vijay Saraswat, Philip J Scott, and RAG Seely. A hyperdoctrinal view of concurrent constraint programming. In *Workshop of Semantics: Foundations and Applications, REX*, pages 457–476. Springer, 1993.

[PU11] Iain Phillips and Irek Ulidowski. A logic with reverse modalities for history-preserving bisimulations. In *Proceedings of the 18th International Workshop on Expressiveness in Concurrency, EXPRESS'11*, pages 104–118, 2011.

[PV15] Salim Perchy and Frank D. Valencia. Opinions and beliefs as constraint system operators. In *Technical Communications of the 31st International Conference on Logic Programming, ICLP 2015*, 2015.

[Rét98] Jean-Hugues Réty. Distributed concurrent constraint programming. *Fundamenta Informaticae*, pages 323–346, 1998.

[RS97] Mark Ryan and Pierre-Yves Schobbens. Counterfactuals and updates as inverse modalities. *Journal of Logic, Language and Information*, pages 123–146, 1997.

[Ryb97] Vladimir V Rybakov. *Admissibility of logical inference rules*, volume 136. Elsevier, 1997.

[SCH10] Chiaki Sakama, Martin Caminada, and Andreas Herzig. A logical account of lying. In *Proceeedings of the 12th European Conference of Logics in Artificial, JELIA 2010*, pages 286–299. Springer, 2010.

[SRP91] Vijay A. Saraswat, Martin Rinard, and Prakash Panangaden. Semantic foundations of concurrent constraint programming. In *Conference Record of the Eighteenth Annual ACM Symposium on Principles of Programming Languages*, pages 333–352, 1991.

[VD14] Hans Van Ditmarsch. Dynamics of lying. *Synthese*, pages 745–777, 2014.

[VDVESW12] Hans Van Ditmarsch, Jan Van Eijck, Floor Sietsma, and Yanjing Wang. On the logic of lying. In *Games, actions and social software*, pages 41–72. Springer, 2012.

[Vic96] Steven Vickers. *Topology via logic*. Cambridge University Press, 1st edition, 1996.

Index

Titre : Opinions, Mensonges et Connaissance. Une Approche Algébrique à la Mobilité de l'Information et des Processus.
Mots clés : Restrictions, Applications Mobiles, Logique Épistémique, Detection de Mensonges, Théorie de Treillis.

Résumé : Système de contraintes (cs – selon l'acronyme anglais) sont formalismes déclaratifs de la théorie de la concurrence tels que les algèbres de processus (p. ex. ccp). Les cs sont souvent représentés par des treillis : ses éléments, appelées contraintes, représentent des informations partiales tandis que l'ordre du treillis correspond à des implications. Récemment, une notion appelée "système de contraintes spatiales à n-agents" a été développée pour représenter l'information dans les systèmes multi-agents et spatialement distribués. Un cs spatial peut être utilisé pour spécifier l'information partiale contenue dans l'espace d'un certain agent (information locale), ou d'un point de vue épistémique, l'information qui est considérée vrai pour un certain agent (croyance). Les cs spatiales, néanmoins, ne fournissent pas de mécanismes pour la spécification de la mobilité de l'information ou des processus.

Dans cette thèse nous avons développé la théorie des systèmes de contraintes spatiales avec des opérateurs pour spécifier le déplacement des informations et processus entre les espaces. Nous étudions les propriétés de cette nouvelle famille de cs et nous illustrons ses applications. Ces nouveaux opérateurs nous apportent de l'extrusion d'informations/processus, et du point de vue épistémique, l'extrusion correspond à ce que nous appelons énonciation ; une information qu'un agent souhaite communiquer à d'autres mais qui peut être inconsistante avec ses croyances. Des énonciations peuvent donc être utilisées pour exprimer des notions épistémiques tels que les canulars ou les mensonges.

Globalement, les cs peuvent exprimer la croyance/énonciation et la connaissance en utilisant respectivement une paire de fonctions espace/extrusion, et un opérateur spatial dérivé qui représente l'information globale. Par ailleurs, nous montrons qu'en utilisant un type précis de nos cs nous pouvons aussi représenter la notion du temps comme une séquence d'instances.

Title: Opinions, Lies and Knowledge. An Algebraic Approach to Mobility of Information and Processes.
Keywords: Constraint Systems, Mobility, Epistemic Logic, Lies, Concurrency Theory.

Abstract: Constraint systems (cs) are declarative formalisms from concurrency theory such as process calculi (e.g. ccp). Cs are often represented as lattices: their elements, called constraints, represent partial information and their order correspond to entailment. Recently a notion of n-agent spatial cs was introduced to represent information in spatially distributed multi-agent systems. A spatial cs can be used to specify partial information holding in a given agent's space (local information), or from an epistemic point of view, a piece of information the agent considers true (beliefs). Spatial cs however do not provide mechanisms for specifying the space mobility of information/processes.

In this thesis, we develop the theory of spatial cs with operators to specify information and processes moving between spaces. We investigate the properties of this new family of cs and illustrate their applications. The new operators provide for process/information extrusion, and from an epistemic point of view, said extrusion corresponds to what we shall called utterance; information that an agent communicates to others but that may be inconsistent with the agent's beliefs. Utterances can express epistemic notions such as hoaxes, opinions and lies.

On the whole, cs can express the epistemic notions of belief/utterance and knowledge by means of, respectively, a space/extrusion function pair and a derived spatial operator that specifies global information. We shall also see that, by using a specific construction of our constraint systems, we can encode the notion of *time* as an arbitrary nesting of spaces representing a *sequence of instances*.