Belief, Knowledge, Lies and Other Utterances in an Algebra for Space and Extrusion

Michell Guzmán, Stefan Haar, Salim Perchy, Camilo Rueda, Frank Valencia

Belief, Knowledge, Lies and Other Utterances in an Algebra for Space and Extrusion ☆

Michell GUZMAN[a], Stefan HAAR[b], Salim PERCHY[c], Camilo RUEDA[d], Frank D. VALENCIA[e]

[a] *INRIA & LIX École Polytechnique de Paris*

[b] *CNRS & LSV École Normale Supérieur - Cachan*

[c] *INRIA & LIX École Polytechnique de Paris*

[d] *Pontificia Universidad Javeriana de Cali*

[e] *CNRS-LIX École Polytechnique de Paris & Pontificia Universidad Javeriana de Cali*

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 article we develop the theory of spatial constraint systems with operators to specify information and processes moving from a space to another. We shall investigate the properties of this new family of constraint systems and illustrate their applications. From a computational point of view the new operators

☆This work has been partially supported by the ANR project 12IS02001 PACE, the Colciencias project 125171250031 CLASSIC, and Labex DigiCosme (project ANR-11-LABEX-0045-DIGICOSME) operated by ANR as part of the program "Investissement d'Avenir" Idex Paris-Saclay (ANR-11-IDEX-0003-02)

Email addresses: michell.guzman@inria.fr (Michell GUZMAN), stefan.haar@inria.fr (Stefan HAAR), yamil-salim.perchy@inria.fr (Salim PERCHY), camilo.rueda@cic.puj.edu.co (Camilo RUEDA), frank.valencia@lix.polytechnique.fr (Frank D. VALENCIA)

provide for process/information *extrusion,* a central concept in formalisms for *mobile communication.* From an epistemic point of view extrusion corresponds to a notion 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 then be used to express instances of epistemic notions such as *hoaxes* or intentional *lies.* Spatial constraint system can express the epistemic notion of *belief* by means of space functions that specify *local information.* We shall show that spatial constraint can also express the epistemic notion of knowledge by means of a derived spatial operator that specifies *global information.*

Keywords: Order Theory, Algebraic Structures, Concurrency Theory, Epistemic Logic, Modal Logic, Lattices, Space, Extrusion, Mobility, Belief, Lies.

Contents

1. Introduction

Epistemic, mobile and spatial behavior are common place in today's distributed systems. The intrinsic *epistemic* nature of these systems arises from social behavior. Most 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 those decisions' dependance on the results of interactions with others within some 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 that sprang from the interactions with other participants involved in that situation. Appropriate performance within 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. We therefore 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) [39] were designed to give explicit access to the concept of partial information and, as such, have close ties with logic [33, 30]. This makes them ideal for the incorporation of epistemic and spatial concepts by expanding the logical connections to include *multi-agent modal logic* [27]. In fact, the sccp calculus [26] extends ccp with the ability to define local computational spaces where agents can store epistemic information and run processes.

Problem: Spatial and Epistemic Mobility

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.

Constraint Systems. The notion of *constraint system (cs)* is central to ccp and other declarative formalisms such as (concurrent) constraint logic programming (clp). All ccp calculi 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 [39].

Similarly, the notion of computational space and the epistemic notion of belief in sccp [26] 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. that 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.

Algebraic Structures for Extrusion and Epistemic Reasoning

Our main goal in this article is to investigate algebraic operations in spatial constraint systems that help provide semantic foundations to reason about extrusion and epistemic phenomena. From a computational point of view, the new operations will allow us to specify mobile behavior as constraints. From a logic point of view, they will allow us to specify epistemic concepts such as belief, knowledge, utterances, opinions, and intentional lies.

Contributions

In this article 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 original spatial theory of sccp [26], which captures belief, can also capture an epistemic notion of knowledge. This latter contribution does not involve extrusion but it is consistent with our goal of using algebraic spatial structures to capture epistemic behaviour.

Our main contributions can be summarized and structured as follows.

1. *Extrusion 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 [12]. 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. *The 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. *Properties 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. *Application: 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 [26]. 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. *Knowledge 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 [26] 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.

This submission is the revised and extended version of [23] for the special JLAMP issue of papers presented at ICE 2015. In this version we have included proofs and new material corresponding to our spatial interpretation of knowledge in terms of global space described in the last point of the above-mentioned contributions. This new material, although it is orthogonal to the issue of extrusion, complements the other contributions: We shall express meaningful epistemic notions as spatial concepts: Belief as a *local space*, knowledge as a *global space*, and lies and other utterances as *extrusion* from a local space.

Organization

This article is structured as follows. In Section 2 we recall the notions of constraint system (cs) and spatial cs (scs). In Section 3 we introduce scs with extrusion (scse) and illustrate spatial and epistemic specifications. The Extrusion problem is given in Section 3.4 and the properties of space and extrusion are stated in Section 3.5. In Section 4 we derive a logic of belief and utterance as an application of the results stated in previous sections. Finally, Section 5 provides a representation of knowledge in terms of a derived spatial operation for global information.

For convenience, this paper includes an index table for notation.

2. Background on Constraint Systems

In this section we recall the notion of basic constraint system and the more recent notion of spatial constraint system [26]. We presuppose basic knowledge of order theory and modal logic [1, 35, 18, 5].

2.1. Plain Constraint Systems

The ccp model is parametric in a *constraint system* (cs) specifying the structure and interdependencies of the partial information that processes can ask of and post in a *shared store*. This information is represented as *assertions* traditionally referred to as *constraints*.

Following [6] we formalize constraint systems as *complete algebraic lattices* (an alternative syntactic characterization of cs, akin to Scott information systems, is given in [39, 33]). 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 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.

Definition 2.1 (Constraint Systems [6])**.** *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* [42].

Let us now recall some notions and notation from order theory.

Notation 2.1 (Lattices)**.** *Let $\mathcal{C}$ be a partially ordered set (poset) $(Con, \sqsubseteq)$. We shall use $\bigsqcup S$ to denote the least upper bound (lub) (or* supremum *or* join*) of the elements in S, and $\bigsqcap S$ is the greatest lower bound (glb) (*infimum *or* meet*) of the elements in S. We say that $\mathcal{C}$ is a* complete lattice *iff each subset of Con has a supremum and a infimum in Con. A non-empty set $S \subseteq Con$ is* directed / filtered *iff every* finite *subset of S has an upper bound / lower bound in S. Also $c \in Con$ is* compact (or finite) *iff for any directed subset D of Con, $c \sqsubseteq \bigsqcup D$ implies $c \sqsubseteq d$ for some $d \in D$. A complete lattice $\mathbf{C}$ is said to be* algebraic *iff for each $c \in Con$, the set of compact elements below it forms a directed set and the lub of this directed set is c.*

We conclude this section by briefly describing two typical concrete constraint systems.

Example 2.1 (Herbrand Constraint System [6, 39])**.** *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.* □

In the above example constraints are represented as set of equations and thus the join (lub) of constraints corresponds to the equivalent class of *union* of their equations. We can also view a constraint c as a representation of a set of variable assignments [2]. 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 (Boolean Constraint System [2])**.** *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

$$\phi, \psi, \ldots \ := \ p \mid \phi \wedge \psi \mid \neg\phi \tag{2.1}$$

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 $\neg\mathrm{F}$. 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 $\mathbf{B}[\![\phi]\!] \stackrel{\mathtt{def}}{=} \{\pi \in \mathcal{A}(\Phi) \mid \pi \models \phi\}$ 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.

Other typical examples include constraint system for streams (the Kahn cs), rational intervals, and first-order theories [39].

2.2. Spatial Constraint Systems

The authors of [26] 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

$$[c]_i \tag{2.2}$$

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

$$[[c]_j]_i \tag{2.3}$$

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

$$[c]_i \sqcup [d]_j \tag{2.4}$$

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 (Spatial Constraint System [26])**.** *An n-agent* spatial constraint system (n-scs) $\mathbf{C}$ *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.

A concrete instance of spatial constraint systems will be given in Definition 2.5. We now give some intuition about the space properties. Property S.1 in Definition 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 2.1 (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 2.3 (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:*

$$[c]_G \stackrel{\texttt{def}}{=} \bigsqcup_{i \in G} [c]_i \quad \textit{and} \quad [\![c]\!]_G \stackrel{\texttt{def}}{=} \bigsqcup_{j=0}^{\infty} [c]_G^j \tag{2.5}$$

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 [18].

Kripke Spatial Constraint Systems. We conclude this section with a concrete spatial constraint system from [26]. This constraint system will play a significant role later in Section 4. We basically extend Example 2.2 by moving from Boolean assignments to *Kripke structures*. Other examples of spatial constraint system for epistemic reasoning are Aumann structures [26].

Definition 2.4 (Kripke Structures)**.** *An n-agent Kripke structure (model) (KS) M over a set of atomic propositions Φ is a tuple:*

$$M = (S, \pi, \mathcal{R}_1, \ldots, \mathcal{R}_n) \tag{2.6}$$

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 2.2. *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. 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.*

Recall that in Example 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 2.5 (Kripke scs [26])**.** *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*

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

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

$$[X]_i = i(X) \text{ where } i(X) \stackrel{\texttt{def}}{=} \{(M, s) \in \Delta \mid \forall t : s \xrightarrow{i}_M t \text{ implies } (M, t) \in X\} \tag{2.7}$$

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, a constraint c in $\mathbf{K}(\mathcal{S}_n(\Phi))$ is compact iff $\Delta \setminus c$ is a finite set [26].

Proposition 2.1. *Let $\mathbf{K}(\mathcal{S}_n(\Phi)) = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$ 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:

$[\mathit{true}]_i$ 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 $[\mathit{true}]_i = \mathit{true}$.

For S.2:

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

□

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 used in Section 4.

The modal language $\mathcal{L}_n(\Phi)$ is obtained by extending the grammar for the propositional language $\mathcal{L}_0(\Phi)$ in Equation 2.1 with modalities $\Box_i\phi$ in the standard way.

Definition 2.6 (Modal Language)**.** *Let Φ be a set of primitive propositions. The language $\mathcal{L}_n(\Phi)$ is given by the following grammar:*

$$\phi, \psi, \ldots \ := \ p \mid \phi \wedge \psi \mid \neg\phi \mid \Box_i\phi \tag{2.8}$$

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

The semantics of modal logics is typically given using KS's. We say that a pointed KS (M, s) *satisfies* ϕ iff $(M, s) \models \phi$ where $\models$ is defined 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 $(M, s) \not\models \phi$, and $(M, s) \models \Box_i\phi$ iff $(M, t) \models \phi$ for every t such that $s \xrightarrow{i}_M t$.

As in Example 2.2 we can interpret each formula ϕ as constraints in Kripke constraint systems.

Definition 2.7 (Kripke Constraint Interpretation)**.** *Let $\mathcal{C}$ be a Kripke scs $\mathbf{K}(\mathcal{S}_n(\Phi))$. Given a modal formula ϕ in the language $\mathcal{L}_n(\Phi)$, its interpretation in the Kripke scs $\mathcal{C}$ is the constraint $\mathcal{C}[\![\phi]\!]$ inductively defined as follows:*

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

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

Notation 2.3. *Notice that the interpretation of $\Box_i(\phi)$, $\mathcal{C}[\![\Box_i(\phi)]\!]$, is equal to the constraint $[\mathcal{C}[\![\phi]\!]]_i$ in $\mathbf{K}(\mathcal{S}_n(\Phi))$. 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.

Remark 2.2 (Boolean Implication)**.** *Constraint systems of the form $(\mathcal{P}(U), \supseteq)$, as $\mathbf{B}(\Phi)$ in Example 2.2 and $\mathbf{K}(\mathcal{S}(\Phi))$ in Definition 2.5, are standard examples of Boolean algebras [20]. 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.*

3. Spatial Constraint Systems with Extrusion

This is the main section of this article. We shall introduce a new notion 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 systems to scse's. We will then state some distinctive properties of space and extrusion as well as a correspondence between these two concepts that will be used later on in Section 4.

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:

$$\text{E.1: } [\uparrow_i c]_i = c. \tag{3.1}$$

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 (Spatial Constraint System with Extrusion)**.** *An n-agent* spatial constraint system with extrusion (n-scse) $\mathbf{C}$ *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 $\mathbf{C}$ 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 3.4.2. In the next section 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 (Section 2) we discussed an interpretation of implication that works for Boolean and Kripke cs. We can define a general form of implication by adapting the corresponding notion from Heyting Algebras [42] 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 (Heyting Implication and Negation)**.** *Let $\mathcal{C}$ be a constraint system $(\mathit{Con}, \sqsubseteq)$. Define $c \to d$ as:*

$$\bigsqcap\{e \mid e \sqcup c \sqsupseteq d\} \tag{3.2}$$

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

The above construction corresponds to (intuitionistic) implication in lattices that are *frames* [42].

Definition 3.3 (Frames)**.** *A cs $(\mathit{Con}, \sqsubseteq)$ is said to be a* frame *iff its joins distribute over arbitrary meets: More precisely, $c \sqcup \bigsqcap S = \bigsqcap\{c \sqcup e \mid e \in S\}$ for every $c \in \mathit{Con}$ and $S \subseteq \mathit{Con}$.*

Remark 3.1. *The Boolean constraint system and the family of Kripke constraint systems (Examples 2.2 and Definition 2.5) are all frames 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., $\mathbf{B}(\Phi)$ in Example 2.2 and $\mathbf{K}(\mathcal{S}(\Phi))$ in Definition 2.5, the operators $\to$ and $\sim$ are known to coincide with the constructions $\Rightarrow$ and $\neg$ defined in Remark 2.2, since boolean implication and boolean negation are particular cases of Heyting implication and Heyting negation [42].*

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 Example 2.1 with variables x and y and constants 0 and 1. Consider the constraints $a = \{x = 0\}, b = \{x = 0, y = 0\}$ and $c = \{x = 1\}$. We have

$$a \sqcup (b \sqcap c) = a \sqcup \mathit{true} = a \neq (a \sqcup b) \sqcap (a \sqcup c) = b \sqcap \mathit{false} = b.$$

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

Lemma 3.1 (Modus-Ponens)**.** *Suppose that* $(Con, \sqsubseteq)$ *is a frame. Then for every* c, d*, we have:*

$$c \sqcup (c \to d) = c \sqcup d \tag{3.3}$$

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 $c \sqcup d \sqsubseteq c \sqcup c \sqcup \bigsqcap S = c \sqcup \bigsqcap S$.

□

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

Remark 3.2. *For the applications examples in this section, 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 [41]. 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 3.1 (Hoax)**.** *Suppose that* $c \sqcup d = \mathit{false}$*. The assertion*

$$[c \sqcup \uparrow_i d]_i \tag{3.4}$$

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 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) followed by the extrusion axiom E.1 (Definition 3.1) 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 3.4 to $[c]_i \sqcup d$ *with* $c \sqcup d = \mathit{false}$ *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 (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*

$$[\sim c \to \uparrow_j d]_j. \tag{3.5}$$

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

$$[c \sqcup \uparrow_i [\sim c]_j]_i. \tag{3.6}$$

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 3.5 and 3.6, we derive the expected result.

$$\begin{aligned}
&[\sim c \to \uparrow_j d]_j \sqcup [c \sqcup \uparrow_i [\sim c]_j]_i \\
&= [\sim c \to \uparrow_j d]_j \sqcup [c]_i \sqcup [\uparrow_i [\sim c]_j]_i && \textit{(S.2 on } [\cdot]_i\textit{)} \\
&= [\sim c \to \uparrow_j d]_j \sqcup [c]_i \sqcup [\sim c]_j && \textit{(E.1 on } [\cdot]_i\textit{)} \\
&= [\sim c \sqcup \sim c \to \uparrow_j d]_j \sqcup [c]_i && \textit{(S.2 on } [\cdot]_j\textit{)} \\
&= [\sim c \sqcup \uparrow_j d]_j \sqcup [c]_i && \textit{(Lemma 3.1)} \\
&= d \sqcup [\sim c]_j \sqcup [c]_i && \textit{(E.1 on } [\cdot]_j\textit{)}
\end{aligned}$$

□

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

$$[e \sqcup \uparrow_i (c \to [d]_i)]_i. \tag{3.7}$$

Equation 3.7 specifies the sending of a process $c \to [d]_i$ to the outside of a 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 3.1 we can

derive $[e \sqcup d]_i$ *from* $c \sqcup [e \sqcup \uparrow_i c \to [d]_i]$ *as follows:*

$$\begin{array}{lll} c \sqcup [e \sqcup \uparrow_i (c \to [d]_i)]_i & & \\ = c \sqcup [e]_i \sqcup [\uparrow_i (c \to [d]_i)]_i & & \textit{(S.2)} \\ = c \sqcup [e]_i \sqcup c \to [d]_i & & \textit{(E.1)} \\ = c \sqcup [e]_i \sqcup [d]_i & & \textit{(Lemma 3.1)} \\ = c \sqcup [e \sqcup d]_i & & \textit{(S.2)} \end{array}$$

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

$$[e \sqcup \uparrow_i [c \to \uparrow_j [d]_i]_j]_i \tag{3.8}$$

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 3.1 we obtain the following:

$$[e \sqcup \uparrow_i [c \to \uparrow_j [d]_i]_j]_i \sqcup [c]_j \quad \sqsupseteq \quad [e \sqcup d]_i. \tag{3.9}$$

(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 3.4 (Global Extrusion)**.** *Let* $\mathcal{C}$ *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* $\mathcal{C}$ *are defined as:*

$$\uparrow_G c \stackrel{\texttt{def}}{=} \bigsqcup_{i \in G} \uparrow_i c \quad \textit{and} \quad \Uparrow_G c \stackrel{\texttt{def}}{=} \bigsqcup_{j=0}^{\infty} \uparrow_G^j c \tag{3.10}$$

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 2.3. 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 previous 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 3.4 (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 2.3 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

$$e \sqsupseteq [[\ldots[c]_{i_m}\ldots]_{i_2}]_{i_1} \tag{3.11}$$

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

$$e \sqcup [\![c \to \Uparrow_G d]\!]_G \sqsupseteq d. \tag{3.12}$$

□

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*. A space function that preserves the supremum/infimum of any arbitrary subset of *Con* is said to be *join-complete/meet-complete*.

The join-completeness of space functions trivially implies their *(Scott) continuity*, a central concept in domain theory. A space function in $\mathbf{C}$ is *continuous/downward continuous* if it preserves the supremum/infimum of any directed set/filtered set. From S.2 and the fact that constraint systems are complete lattices, the reverse implication is also true: Space continuity implies space completeness.

Proposition 3.1. *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 [20]. □

3.4. The Extrusion Problem.

Given an scs a legitimate question is whether it can be extended to an scse. For instance, we may wonder if Kripke constraint systems (Example 2.4) can be extended with extrusion. 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 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 Lemma 3.2 by showing something stronger: Fibers are closed under finite joins.

Lemma 3.2 (Directed Fibers)**.** *Let* $\mathbf{C}$ *be an scs and let* $[\cdot]_i$ *be a surjective space function of* $\mathbf{C}$*. The fiber of any constraint* c *of* $\mathbf{C}$ *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 3.2 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 3.1 (Max Extrusion)**.** *Let* $\mathbf{C}$ *be an scs and let* $[\cdot]_i$ *be a surjective and continuous space function of* $\mathbf{C}$*. Then* $\uparrow_i : c \mapsto \bigsqcup [c]_i^{-1}$ *is a right inverse of* $[\cdot]_i$*.*

Proof. It follows from Lemma 3.2 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.

3.4.1. 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 3.1 *may* result in $\uparrow_i true \neq true$ for some scs's. To better illustrate this situation, consider the following example.

Example 3.5. *Let* $Con = \mathbb{N} \cup \{\infty\}$ *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* $(Con, \sqsubseteq, [\cdot]_1)$ *is an scs with* $true = 0$. *We can apply Theorem 3.1 to obtain the extrusion function* $\uparrow_1 : c \mapsto \bigsqcup [c]_1^{-1}$ *for* $c \in Con$. *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 3.2. *Let* $\mathbf{C}$ *be an scse with constraints in* Con, *and let* $\uparrow_i$ *be the extrusion function of the agent* i *in* $\mathbf{C}$. *Then*

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

Proof.

First we prove $\uparrow_i true \approx_i true$.

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

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.

3.4.2. 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 3.5 (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 3.1: 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 3.2 states this impossibility result.

Theorem 3.2 (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 3.2 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 1. Let f be the self-map given by the arrows in Figure 1. 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 1 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

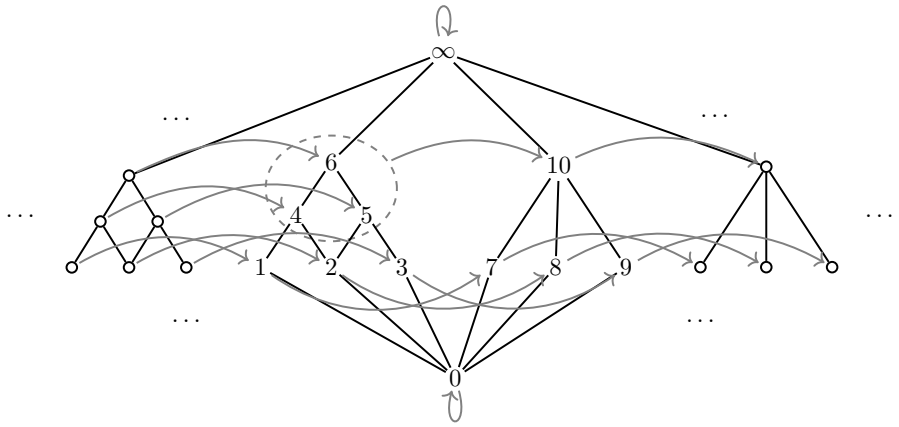

Figure 1: A one-agent scs. Gray arrows depict a surjective and continuous space function over $\mathbb{N} \cup \{\infty\}$.

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 3.2.

Our strategy to prove Theorem 3.2 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 3.3 (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 $\uparrow_i true = true$. From S.1 $true \in [true]_i^{-1}$ thus $\bigsqcap [true]_i^{-1} = true$.

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 $\uparrow_i c \not\sqsupseteq \uparrow_i d$. 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

$$\uparrow_i c \sqcap \uparrow_i d \in [c]_i^{-1}. \tag{3.13}$$

  From Equation 3.13 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 3.3. *Notice that from Proposition 3.1, Theorem 3.1 and Theorem 3.3 are dual in the sense that whereas Theorem 3.1 requires space functions to be* join-complete*, Theorem 3.3 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 3.1. *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. □

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 3.1 (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 the following notions from order theory [14]. 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.

Definition 3.6. *Given* $(\mathit{Con}, \sqsubseteq)$ *a self-map* f *over Con is said to be an* order-embedding *iff* f *preserves and reflects* $\sqsubseteq$*: i.e, for each* $c, d \in \mathit{Con}$: $c \sqsubseteq d$ *implies* $f(c) \sqsubseteq f(d)$ *(*order-preserving*) and* $f(c) \sqsubseteq f(d)$ *implies* $c \sqsubseteq d$ *(*order-reflecting*). Furthermore,* f *is said to be an* order-automorphism *if it is a surjective order-embedding. Finally, 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 3.2 (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 2.1) 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 3.3 (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 3.3 (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 3.1), and by hypothesis it is injective, then it is also a bijection. Any space function is order-preserving since they preserve binary joins (Remark 2.1). 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 3.3 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 [22].

Corollary 3.2 (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 3.3, Corollary 3.2, and Property 3.3 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 2.1). One may then wonder if space functions from scse's are already continuous. A negative answer is given in the example below.

Example 3.6 (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 3.3 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 3.3 to derive an extrusion function for a rather simple scs. Notice that we could not have applied Theorem 3.1 because the $[\cdot]_1$ was shown not to be continuous. In Section 4 we will derive extrusion functions for a meaningful and more involved scs using Theorem 3.1.

3.6. Galois Connections

We conclude this section by stating a pleasant correspondence between space and extrusion. In Example 3.6 we used Theorem 3.3 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.

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 [1].

Property 3.4 (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 [1] 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 3.4 that the pair $(\uparrow_1, [\cdot]_1)$ in Example 3.6 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 3.4.

Example 3.7. *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 3.4 we can conclude that* $(\uparrow_i, [\cdot]_i)$ *is not a Galois connection. (One can also verify using Property 3.4 that the reversed pair* $([\cdot]_i, \uparrow_i)$ *is not a Galois connection either.)* □

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

$$[c]_i \sqsubseteq d \Leftrightarrow c \sqsubseteq \uparrow_i d \tag{3.14}$$

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 conclude this section, the main and longest of this paper, with a summary of its results.

We considered the problem of deriving extrusion for space functions. 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 3.1 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 3.3 gives us a dual construction that maps each constraint to the greatest lower bound of its pre-image under the space function.

In the next section we will use some of the above results to derive extrusion for a modal (epistemic) logic of belief. For modal logics in general, the underlying constraint system is Kripke's (Example 2.5) and thus space functions correspond to the box ($\Box$) operator, the join correspond to conjunction, and meet correspond to disjunction (see Definition 2.7). 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 3.1 always applies since the space functions of any Kripke constraint system are join-complete. In fact the box operator always distributes over conjunction but not always over disjunction.

Nevertheless, there are cases where space functions are also meet-complete, for example when the box operator is interpreted as the *next modality* $\bigcirc$ of temporal logic [28] since this modality distributes over both conjunction and disjunction[1]. In this case we could apply both Theorem 3.1 and Theorem 3.3. Interestingly, we would obtain two different but well-known reverse modalities for the next operator. Theorem 3.1 would give us a *strong-previous* modality $\ominus$ while Theorem 3.3 would give us a *weak-previous* modality $\widetilde{\ominus}$. Notice that unlike Theorem 3.1, Theorem 3.3 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).

Finally, we presented distinctive properties of space and extrusion. Property 3.1 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 3.3 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.

4. Epistemic Applications: Kripke SCS with Extrusion & Belief with 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 2.5) as an application of the results we obtained in Sections 3.4 and 3.5.

[1]We shall briefly discuss this issue to clarify the use of Theorem 3.1 and Theorem 3.3. However, the technical development is out of the scope of this paper.

Recall that we can associate a modal language (Definition 2.6) to a Kripke scs by interpreting formulae as constraints, i.e., set of pointed Kripke structures (KS's). Under such association, $\Box_i\phi = [\phi]_i$ states that ϕ holds true in the space of i (Notation 2.3). 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 2.5. 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 :

$$[c]_i \stackrel{\texttt{def}}{=} \{(M, s) \in \Delta \mid \forall t : \text{ if } s \xrightarrow{i}_M t \text{ then } (M, t) \in c\} \tag{4.1}$$

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

We will thus endeavour 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 3.1 to derive an extrusion for each space function over the restricted KS's.

4.1. Left-total left-unique Kripke Structures

Modal logics are typically interpreted over different families of KS's obtained by imposing conditions on their accessibility relations. (E.g, if the intended meaning of the modality $\Box_i(\phi)$ is the knowledge of a fact ϕ by agent i then the accessibility relations ought to be equivalence relations.) For the weakest modal logic (the system K_n) there are no conditions on the accessibility relation 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 4.1. *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 3.1 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 2.2 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 4.1 we conclude that $(M', s') \in [\mathit{false}]_i$ thus violating space consistency. Property 3.1 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 3.1, 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 4.1 (Left-total space-consistency)**.** $\mathcal{M}^{lt}$ *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 $\mathcal{S}$ 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 3.1).

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 $c \in \mathcal{P}(\mathcal{M}^{\mathtt{lt}})$ such that $c = [d]_i$ for some $d \in \mathcal{P}(\mathcal{M}^{\mathtt{lt}})$. Since $\mathcal{W}_i(M, s) = \mathcal{W}_i(M, s')$, from Equation 4.1 we conclude that if $(M, s) \in c$ then $(M, s') \in c$. Thus, surjectivity is not satisfied by $[\cdot]_i$ since for every $d \in \mathcal{P}(\mathcal{M}^{\mathtt{lt}})$, $[d]_i \neq \{(M, s)\}$. 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 4.2 (Left-unique surjectivity)**.** *The set* $\mathcal{M}^{ltu}$ *satisfies surjectivity.*

Proof. It follows from Claim C.1 in the proof of Lemma 4.1. □

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 3.1 and 3.3.

We cannot apply Theorem 3.3 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. *The space functions of* $\mathbf{K}(\mathcal{M}^{ltu})$ *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 $S \subseteq \mathcal{P}(\mathcal{M}^{\mathtt{ltu}})$ 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 3.1 and derive the following extrusion function for each $[\cdot]_i$ in $\mathbf{K}(\mathcal{M}^{\mathtt{ltu}})$:

$$\uparrow_i : c \mapsto \bigsqcup [c]_i^{-1}. \tag{4.2}$$

Furthermore, we can show that the construction in Equation 4.2 is equivalent to the intensional definition given below.

Lemma 4.1. *(Extrusion for Kripke Spaces). Let* $\uparrow_i$ *be defined as in Equation 4.2 over the Kripke scs* $\mathbf{K}(\mathcal{M}^{ltu})$*. Then*

$$\uparrow_i(c) = i^{-1}(c) \text{ with } i^{-1}(c) \stackrel{\mathtt{def}}{=} \{(M,t) \in \Delta \mid \exists s :\ s \xrightarrow{i}_M t \text{ and } \ (M,s) \in c\} \tag{4.3}$$

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

Proof. Let

$$\uparrow'_i c = \{(M,t) \in \Delta \mid \exists s :\ s \xrightarrow{i}_M t \text{ and } \ (M,s) \in c\}. \tag{4.4}$$

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 4.4, 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 **Claim 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 4.4 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 4.1 (Kripke scs with extrusion)**.** *The n-agent scs with extrusion $\mathbf{K}^{\uparrow}(\mathcal{M}^{ltu})$ results from extending $\mathbf{K}(\mathcal{M}^{ltu})$ with an extrusion function $\uparrow_i$ for each $i \in \{1,\ldots,n\}$ defined as in Equation 4.2.*

It follows from Property 3.4 that in the derived scse, space and extrusion form a *Galois connection*.

Corollary 4.1. *Let $[\cdot]_i$ and $\uparrow_i$ be the space and extrusion function of agent i in $\mathbf{K}^{\uparrow}(\mathcal{M}^{ltu})$. The pair $([\cdot]_i, \uparrow_i)$ is a Galois connection.*

We shall apply this Galois connection in the following section.

4.2. The BU_n logic

We shall now extend the modal language in Definition 2.6 with modalities to express utterances. The intended meaning and properties of the formulae in the extended language will be given from the scse $\mathbf{K}(\mathcal{M}^{\mathtt{ltu}})$ we derived in the previous section (Definition 4.1). 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 2.5 and extending it with modalities U_i.

Definition 4.2 (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 $i \in \{1 \dots n\}$ 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 2.2 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. 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 2 illustrates the above-mentioned description.

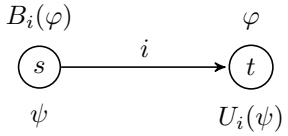

Figure 2: Illustration of $s \xrightarrow{i}_M t$ with $B_i\varphi$ and ψ true at s and φ and $U_i\psi$ true at t

Derived Specifications. We expect the following formula to be valid:

$$B_iU_i\varphi \Leftrightarrow \varphi. \tag{4.5}$$

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:

$$\mathcal{O}_i(\varphi) \Leftrightarrow (B_i\varphi) \wedge \varphi. \tag{4.6}$$

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

$$\mathcal{H}_i(\varphi) \Leftrightarrow (B_i\neg\varphi) \wedge \varphi. \tag{4.7}$$

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:

$$B_i\varphi \Rightarrow \hat{B}_i\varphi \text{ and } U_i\varphi \Rightarrow \hat{U}_i\varphi. \tag{4.8}$$

The formulae in Equation 4.8 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 $\mathbf{K}^{\uparrow}(\mathcal{M}^{\texttt{ltu}})$ in Definition 4.1. Recall our definition of the negation constraint $\sim c$ (Definition 3.2) and that $\mathbf{K}^{\uparrow}(\mathcal{M}^{\texttt{ltu}})$ is also a frame (Remark 3.1).

Definition 4.3. *Let $\mathbf{K}^{\uparrow}(\mathcal{M}^{ltu}) = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n, \uparrow_1, \ldots, \uparrow_n)$ be the scse in Definition 4.1. Given φ in $\mathcal{L}_n^{BU}(\Phi)$, its denotation $\mathbf{K}^{\uparrow}[\![\varphi]\!]$ is inductively defined as follows:*

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

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 $\mathbf{K}^{\uparrow}[\![\varphi]\!] = true$.

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 4.5, 4.6, 4.7 and 4.8.

Proposition 4.4. *The formulae in Equations 4.5, 4.6, 4.7 and 4.8 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, ${\sim}[{\sim}\mathbf{K}^\uparrow[\![\varphi]\!]\ ]_{\mathbf{i}}$ 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 4.1 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 4.2. *$\varphi \Rightarrow B_i\psi$ is valid in BU_n iff $U_i\varphi \Rightarrow \psi$ is valid in BU_n.*

Intuitively Corollary 4.2 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 section by revisiting Example 3.2.

Example 4.1 (Hoax Communication)**.** *Let us consider the epistemic formulae $F = B_j(\neg p \Rightarrow U_j(q))$ and $G = B_i(p \wedge U_iB_j(\neg p))$. 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:

$$
\begin{aligned}
\mathbf{K}^{\uparrow}[\![F \wedge G]\!] &= [\sim \mathbf{K}^{\uparrow}[\![p]\!] \to \uparrow_j \mathbf{K}^{\uparrow}[\![q]\!]]_j \ \sqcup\ [\mathbf{K}^{\uparrow}[\![p]\!] \ \sqcup\ \uparrow_i [\sim \mathbf{K}^{\uparrow}[\![p]\!]]_j]_i && \textit{Def.4.3} \\
&= [\sim \mathbf{K}^{\uparrow}[\![p]\!] \to \uparrow_j \mathbf{K}^{\uparrow}[\![q]\!]]_j \ \sqcup\ [\mathbf{K}^{\uparrow}[\![p]\!]]_i \ \sqcup\ [\uparrow_i [\sim \mathbf{K}^{\uparrow}[\![p]\!]]_j]_i && \textit{S.2}\ [\cdot]_i \\
&= [\sim \mathbf{K}^{\uparrow}[\![p]\!] \to \uparrow_j \mathbf{K}^{\uparrow}[\![q]\!]]_j \ \sqcup\ [\mathbf{K}^{\uparrow}[\![p]\!]]_i \ \sqcup\ [\ \sim \mathbf{K}^{\uparrow}[\![p]\!]]_j && \textit{E.1}\ [\cdot]_i \\
&= [\sim \mathbf{K}^{\uparrow}[\![p]\!] \sqcup\ \sim \mathbf{K}^{\uparrow}[\![p]\!] \to \uparrow_j \mathbf{K}^{\uparrow}[\![q]\!]]_j \ \sqcup\ [\mathbf{K}^{\uparrow}[\![p]\!]]_i && \textit{S.2}\ [\cdot]_j \\
&= [\sim \mathbf{K}^{\uparrow}[\![p]\!] \ \sqcup\ \uparrow_j \mathbf{K}^{\uparrow}[\![q]\!]]_j \ \sqcup\ [\mathbf{K}^{\uparrow}[\![p]\!]]_i && \textit{Lem.3.1} \\
&= \mathbf{K}^{\uparrow}[\![q]\!] \ \sqcup\ [\sim \mathbf{K}^{\uparrow}[\![p]\!]]_j \ \sqcup\ [\mathbf{K}^{\uparrow}[\![p]\!]]_i && \textit{E.1}\ [\cdot]_j \\
&= \mathbf{K}^{\uparrow}[\![q]\!] \ \cap\ [\sim \mathbf{K}^{\uparrow}[\![p]\!]]_j \ \cap\ [\mathbf{K}^{\uparrow}[\![p]\!]]_i && \textit{Def.2.5} \\
&\subseteq \mathbf{K}^{\uparrow}[\![q]\!]
\end{aligned}
$$

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

5. Knowledge in Terms of Global Space

In previous sections we saw how spatial constraint systems can be used to represent epistemic concepts such as beliefs, lies and opinions. In this section we show that spatial constraint systems can also express the epistemic concept of knowledge using the notion of global information from Definition 2.3.

5.1. Knowledge Constraint System.

In [26] the authors extended the notion of spatial constraint system to account for *knowledge*. In this article we shall refer to the extended notion in [26] 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* [29]: i.e., monotone, extensive and idempotent functions that preserve bottom and lubs.

Definition 5.1 (Knowledge Constraint System [26])**.** *An n-agent* S4 constraint system (n-s4cs) $\mathbf{C}$ *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, 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 2.1); 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 (Definition 2.4) obtained by restricting their accessibility relations. We use $\mathcal{M}_n(\Phi)$ to denote the set of *all* Kripke structures over the set of primitive propositions Φ (Notation 4.1). 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 section, 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 2.5). In [26] it was shown that $\mathbf{C}^{\mathtt{rt}}$ is in fact an S4 constraint system.

Proposition 5.1 ([26])**.** $\mathbf{C}^{rt}$ *is an* s4cs.

Recall our interpretation of formulae of the modal language $\mathcal{L}_n(\Phi)$ (Definition 2.6) using Kripke spatial constraint systems (Definition 2.7). 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 [25].

Definition 5.2. *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 $M \in \mathcal{M}^{rt}(\Phi)$ and s is a state of M, we have $(M, s) \models \phi$.*

Notation 5.1. *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 [26] 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 ([26])**.** $\mathbf{C}^{rt}[\![\phi]\!]$ = *true if and only if ϕ is S4-valid.*

The above gives a brief summary of the use in [26] 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 2.3.

5.2. Knowledge as Global Information.

Let $\mathcal{C} = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n)$ be a *spatial* constraint system. From Definition 2.3 we obtain the following equation:

$$[\![c]\!]_{\{i\}} = c \sqcup [c]_i \sqcup [[c]_i]_i \sqcup [[[c]_i]_i]_i \sqcup \ldots = c \sqcup [c]_i \sqcup [c]_i^2 \sqcup [c]_i^3 \sqcup \ldots = \bigsqcup_{j=0}^{\infty} [c]_i^j \quad (5.1)$$

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* [25].

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. *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 5.1. *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 5.1 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

$$[\![c \sqcup d]\!]_i = \bigsqcup_{j=0}^{\infty} [c \sqcup d]_i^j = \bigsqcup_{j=0}^{\infty} ([c]_i^j \sqcup [d]_i^j). \quad (5.2)$$

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 5.1.
- 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 5.1 $[\![c]\!]_i = \bigsqcup S$ 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 2.5). 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 3.1). Recall our definition of the negation constraint $\sim c$ for frames (Definition 3.2).

Definition 5.4. *Let ϕ be a modal formula from the modal language $\mathcal{L}_n(\Phi)$. The interpretation of ϕ in $\mathbf{C}^*$ is inductively defined as follows:*

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

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 5.1

$$\mathbf{C}^*[\![\Box_i\phi]\!] = [\![\ \mathbf{C}^*[\![\phi]\!]\ ]\!]_i = \bigsqcup_{j=0}^{\omega} [\mathbf{C}^*[\![\phi]\!]]_i^j = \bigcap_{j=0}^{\omega} [\mathbf{C}^*[\![\phi]\!]]_i^j \tag{5.3}$$

In particular, notice that from Theorem 5.1 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 section with the following theorem stating the correctness wrt validity of the interpretation of knowledge as as global operator.

Theorem 5.2. *$\mathbf{C}^*[\![\phi]\!] = true$ 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 2.5 and 5.3 we conclude that $\Delta = true$ in $\mathbf{C}^*$ and $\Delta^{\mathtt{rt}} = true$ in $\mathbf{C}^{\mathtt{rt}}$. From Definitions 2.5 and Proposition 5.2, it suffices to prove that

$$\mathbf{C}^*[\![\phi]\!] = \Delta \text{ if and only if } \mathbf{C}^{\mathtt{rt}}[\![\phi]\!] = \Delta^{\mathtt{rt}} \tag{5.4}$$

Property 5.4 is a corollary of the following two properties:

$$\text{For all } (M, s) \in \Delta^{\mathtt{rt}}: \ (M, s) \in \mathbf{C}^{\mathtt{rt}}[\![\phi]\!] \text{ if and only if } (M, s) \in \mathbf{C}^*[\![\phi]\!] \tag{5.5a}$$

$$\text{For all } (M, s) \in \Delta: \ (M, s) \in \mathbf{C}^*[\![\phi]\!] \text{ if and only if } (M^*, s) \in \mathbf{C}^*[\![\phi]\!] \tag{5.5b}$$

Proof of 5.5a. 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 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 2.5 and Equation 5.3 it suffices to show that $(M, s_k) \in \mathbf{C}^*[\![\psi]\!]$ for $k = 0, 1, \ldots$. From the assumption and Definition 2.5 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, $(M,s) \in \bigcap_{j=0}^{\omega}[\mathbf{C}^*[\![\psi]\!]]_i^j$. Then $(M,s) \in [\mathbf{C}^*[\![\psi]\!]]_i$. From Definition 2.5, 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 5.5b. 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 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 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, 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 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.

□

6. Concluding Remarks and Related Work

We introduced the notion of spatial constraint system with extrusion (scse) as complete lattices with self-maps that account for space and extrusion. We regard extrusion functions as the right inverse of space functions. We used scse to model situations with concurrent and epistemic behaviors such as spatial mobility, lies, opinions, belief and utterance. We formalized scse by building upon notions and concepts from order (domain) theory, epistemic (doxastic) theories, and the algebraic treatment of logic in [42, 15]. We also addressed the problem

of constructing extrusion functions for given space function. We also studied properties of space and extrusion such as space consistency, automorphisms and Galois connections.

As an application of the above-mentioned results we derived extrusion functions for an existing spatial constraint system and then used them to give semantics to an epistemic logic of belief and utterance. The two operators were also shown to form a Galois connection. We also showed that spatial constraint system can represent S4 knowledge by using a global space construction.

All in all, we have argued that belief, knowledge and utterances correspond, respectively, to the spatial concepts of local space, global space, and extrusion. We conclude this summary with a table illustrating the correspondence between lattice operators of spatial constraint systems, their instantiation in the Kripke constraint system, and epistemic operators.

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

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

Related and Future Work. Our scse's can be used as constraint systems for concurrent constraint programming (ccp) calculi [39]. 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 [16, 36, 21, 8]. In [36, 16] processes can send constraints using communication channels much like in the π-calculus. Moreover, [21] extends ccp with primitives for process mobility within a hierarchically organized network described as a tree. Additionally, the authors of [8] create an extension to ccp where agents maintain a local store and communicate through a global store. In [31] temporal ccp process can transmit variables using existential and universal quantification. More recently, in [32] 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 [7, 4]. Our approach differs from these works in both conception, technical development, and applications. We view extrusion/utterance as inverses of space/belief and develop this concept using order-theory and epistemic logic. None of the previously-mentioned works is concerned with applications for reasoning about epistemic behaviour.

Epistemic logics have been widely applied to distributed systems; [18] 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 ccp languages.

Inverse modalities have been used in temporal, epistemic and Hennessy-Milner logic. For example, in [37] the logical properties and consequences of introducing inverse modalities in a generic modal logic is thoroughly explained. Also in [34] the authors put forward an extension of Hennessy-Milner logic with a reverse modality for expressing concurrent behavior. In this article, as an application of our general framework of scs with extrusion, we gave semantics to a belief logic with a reverse modality which we called utterance.

Social phenomena such as lies, utterance, opinions have been recently studied in epistemic (doxastic) logic [41, 40, 38]. We follow [41] 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 [41, 40, 38] 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 [41, 38, 40] of the above-mentioned epistemic phenomena can be incorporated in our constraint systems.

Another work that has influenced the design of scs's with extrusion is the ambient calculus [12]. 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., $ac_i \circ [\cdot]_i = id$.

An approach closely related to ours is the spatial logics for concurrency from [11, 10]. 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. It would be interesting to see how a name quantified scs would look and to study the relationship with the framework in [11, 10].

The process calculi in [3, 9, 17] 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.

The question of whether knowledge is definable in terms of belief has played an important role in epistemology. In [24] the authors studied this question in the framework of epistemic and doxastic logics. They proved that epistemic logic S5 cannot/can be explicitly/implicitely 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 S5 while the notion of belief is KD [18]. We plan to address the question of whether S5 knowledge and KD45 belief can be defined in our spatial constraint systems.

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 by allowing an unbounded number of agents.

Galois connections are central to abstract interpretation techniques [13] for the semantics and verification of programs and concurrent systems. We plan to explore the use of the Galois connections introduced in this paper for the semantics and verification of spatially-distributed concurrent ccp systems by building upon the abstract interpretation approach for ccp introduced in [19].

Acknowledgements. We thank Moreno Falaschi, Sophia Knight, Carlos Olarte, and Hans van Ditmarsch for their feedback on earlier versions of this work. We are also indebted to the anonymous reviewers for constructive criticism and invaluable suggestions that help us improve the presentation of this work.

Index

References

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

[2] A. Aristizábal, F. Bonchi, C. Palamidessi, L. 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.

[3] J. Bengtson, M. Johansson, J. Parrow, and B. 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.

[4] S. Bistarelli and F. Santini. A secure non-monotonic soft concurrent constraint language. *Fundamenta Informaticae*, 134:261–285, 2014.

[5] P. Blackburn, M. De Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, 1st edition, 2002.

[6] F. S. Boer, A. Di Pierro, and C. Palamidessi. Nondeterminism and infinite computations in constraint programming. *Theoretical Computer Science*, pages 37–78, 1995.

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

[8] L. Brim, M. Kretinsky, J.-M. Jacquet, and D. Gilbert. Modelling multi-agent systems as synchronous concurrent constraint processes. *Computing and informatics*, 21:565–590, 2002.

[9] M. G. Buscemi and U. 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.

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

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

[12] L. Cardelli and A. 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.

[13] P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth ACM Symposium on Principles of Programming Languages*, pages 238–252, 1977. doi: 10.1145/512950.512973.

[14] B. A. Davey and H. A. Priestley. *Introduction to lattices and order*. Cambridge university press, 2nd edition, 2002.

[15] A. Di Pierro, C. Palamidessi, and F. S. Boer. An algebraic perspective of constraint logic programming. *Journal of Logic and Computation*, pages 1–38, 1997.

[16] J. F. Díaz, C. Rueda, and F. D. Valencia. Pi+- calculus: A calculus for concurrent processes with constraints. *December 1998 Special Issue of Best Papers presented at CLEI'97*, 1998.

[17] F. Fages, P. Ruet, and S. Soliman. Linear concurrent constraint programming: Operational and phase semantics. *Information and Computation*, pages 14–41, 2001.

[18] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning about knowledge*. MIT press Cambridge, 4th edition, 1995.

[19] M. Falaschi, C. Olarte, C. Palamidessi, and F. D. Valencia. Declarative Diagnosis of Temporal Concurrent Constraint Programs. In *ICLP'07*, volume 4670 of *LNCS*, pages 271–285. Springer, 2007.

[20] G. Gierz, K. H. Hofmann, K. Keimel, J. D. Lawson, M. Mislove, and D. S. Scott. *Continuous lattices and domains*. Cambridge University Press, 1st edition, 2003.

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

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

[23] S. Haar, S. Perchy, C. Rueda, and F. 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.

[24] J. Y. Halpern, D. Samet, and E. Segev. Defining knowledge in terms of belief: The modal logic perspective. *The Review of Symbolic Logic*, pages 469–487, 2009.

[25] J. Hintikka. *Knowledge and belief*. Cornell Univeristy Press, 1962.

[26] S. Knight, C. Palamidessi, P. Panangaden, and F. 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.

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

[28] Z. Manna and A. Pnueli. *The temporal logic of reactive and concurrent systems: Specification*. Springer Science & Business Media, 1992.

[29] J. C. C. McKinsey and A. Tarski. The algebra of topology. *Annals of mathematics*, pages 141–191, 1944.

[30] N. P. Mendler, P. Panangaden, P. J. Scott, and R. Seely. A logical view of concurrent constraint programming. *Nordic Journal of Computing*, pages 181–220, 1995.

[31] C. Olarte and F. 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.

[32] C. Olarte, V. Nigam, and E. 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.

[33] P. Panangaden, V. Saraswat, P. J. Scott, and R. Seely. A hyperdoctrinal view of concurrent constraint programming. In *Workshop of Semantics: Foundations and Applications, REX*, pages 457–476. Springer, 1993.

[34] I. Phillips and I. 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.

[35] S. Popkorn. *First steps in modal logic*. Cambridge University Press, 1st edition, 1994.

[36] J. Réty. Distributed concurrent constraint programming. *Fundamenta Informaticae*, pages 323–346, 1998.

[37] M. Ryan and P.-Y. Schobbens. Counterfactuals and updates as inverse modalities. *Journal of Logic, Language and Information*, pages 123–146, 1997.

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

[39] V. A. Saraswat, M. Rinard, and P. 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.

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

[41] H. Van Ditmarsch, J. Van Eijck, F. Sietsma, and Y. Wang. On the logic of lying. In *Games, actions and social software*, pages 41–72. Springer, 2012.

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