An Algebraic View of Space/Belief and Extrusion/Utterance for Concurrency/Epistemic Logic [*]

Stefan HAAR
INRIA & LSV (ENS Cachan & CNRS)
stefan.haar@inria.fr

Salim PERCHY
INRIA & LIX École Polytechnique
yamil-salim.perchy@inria.fr

Camilo RUEDA
Pontificia Universidad Javeriana Cali
camilo.rueda@cic.puj.edu.co

Frank VALENCIA
CNRS & LIX École Polytechnique
Pontificia Universidad Javeriana Cali
frank.valencia@lix.polytechnique.fr

Abstract

We enrich spatial constraint systems with operators to specify information and processes moving from a space to another. We shall refer to these news structures as *spatial constraint systems with extrusion.* 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 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, which are common place in social media, such as hoaxes or intentional lies. Spatial constraint systems with extrusion can be seen as complete *Heyting algebras* equipped with maps to account for spatial and epistemic specifications.

1. Introduction

Motivation. Epistemic, mobile and spatial behavior are commonplace in today's distributed systems. The intrinsic *epistemic* nature of these systems arises from social behavior. We have multiple agents (users) sharing *beliefs*, *opinions* and even intentional *lies* (hoaxes) on social networks. As for the spatial and mobile behavior, compelling examples are provided by apps and data moving across (possibly nested) spaces defined by friend circles, groups, and shared folders in social networks and cloud storage. 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) [27] were designed to give explicit access to the concept of partial information and, as such, had close ties with logic [18, 21]. This makes them ideal for the incorporation of epistemic and spatial concepts by expanding the logical connections to include *multi-agent modal logic* [17]. In fact, the sccp calculus [16] extends ccp with the ability to define local computational spaces where agents can store epistemic information and run processes.

The Problem. Despite being able to express meaningful epistemic and spatial phenomena such as belief, knowledge, 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. To our knowledge, *spatial mobility* can at best be expressed indirectly in sccp or any other ccp process calculus.

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 [27].

[*] This research was partially supported by the ANR 12IS02001 PACE as well as 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)

PPDP'15, July 14-16, 2015, Siena, Italy.
Copyright © 2015 ACM ISBN 978-1-4503-3516-4/15/07…$15.00.
DOI: http://dx.doi.org/10.1145/10.1145/2790449.2790520

Similarly, the notion of computational space and the epistemic notion of belief in sccp [16] correspond to a family of self-maps $[\cdot]_i : Con \to Con$ on the elements of the constraint system Con. These self-maps 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. 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.

Goal. Our goal in this paper is to investigate algebraic operations in the constraint system that provide the semantic foundations for extrusion. 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 utterances, opinions, and intentional lies.

Contributions. In this paper we generalize the underlying theory of spatial constraint systems by adding *extrusion* functions to their structure. These functions provide for the specification of spatial mobility and epistemic concepts such as utterance and lies. Our main contributions can be summarized and structured as follows.

- *Extrusion as the right inverse of space.* We shall first introduce a family of self-maps $\uparrow_i$, 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 [9]. 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.

- *The Extrusion Problem.* We consider the problem of deriving the corresponding extrusion functions $\uparrow_i$ given a cs with space functions $[\cdot]_i$. We will give canonical constructions of extrusion functions as well as impossibility results for their existence for surjective space functions that satisfy certain limit conditions such as Scott-continuity and meet-completeness.

- *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 $[c]_i \sqsubseteq d \Leftrightarrow c \sqsubseteq \uparrow_i d$.

- *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 cs [16]. We also derive the semantics for a logic of belief with reverse modalities by interpreting its formulae as elements in the Kripe cs with extrusion. We can then show how 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.

Organization. This paper 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 3.3 and the properties of space and extrusion are stated in Section 3.4. Finally in Section 4 we derive a logic of belief and utterance as an application of the results stated in previous sections.

2. Constraint Systems

In this section we recall the notion of basic constraint system and the more recent notion of spatial constraint system. We presuppose basic knowledge of order theory and modal logic [1, 4, 13, 23].

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 [5] we formalize constraint systems as *complete algebraic lattices* (an alternative syntactic characterization of cs, akin to Scott information systems, is given in [21, 27]). 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 1 (Constraint Systems [5])**.** *A constraint system (cs)* $\mathbf{C}$ *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* $\mathbf{C}$, *respectively.*

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

Notation 1. *Let* $\mathbf{C}$ *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* *(*infimum *or* meet*) of the elements in S. We say that* $\mathbf{C}$ *is a* complete lattice *iff each subset of Con has a supremum 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 1 (Herbrand cs [5, 27])**.** *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 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.* □

Boolean cs. In the above example constraints are represented as set of equations and thus the join (lub) of constraints corresponds to *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 $x > 42$. 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 (Boolean cs [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 subsets of assignments, $\sqsubseteq$ is $\supseteq$, false is $\emptyset$, true is $\mathcal{A}(\Phi)$, the join $\sqcup$ is $\cap$, and the meet $\sqcap$ 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 $\mathcal{L}(\Phi)$ be the language built from Φ by the grammar

$$\phi, \psi, \ldots \ := \ p \mid \phi \wedge \psi \mid \neg\phi \tag{1}$$

where $p \in \Phi$. We shall use the classical abbreviations $\phi \vee \psi$ for $\neg(\neg\phi \wedge \neg\psi)$, $\phi \Rightarrow \psi$ for $\neg\phi \vee \psi$, 0 for $p \wedge \neg p$, and 1 for $\neg 0$. A boolean assignment π *satisfies* ϕ iff $\pi \models \phi$ where $\models$ is defined inductively as follows: $\pi \models p$ iff $\pi(p) = 1$, $\pi \models \phi \wedge \psi$ iff $\pi \models \phi$ and $\pi \models \psi$, and $\pi \models \neg\phi$ iff $\pi \not\models \phi$. We interpret each formula ϕ as the constraint $\mathbf{B}[\![\phi]\!] \stackrel{\text{def}}{=} \{\pi \in \mathcal{A}(\Phi) \mid \pi \models \phi\}$ in $\mathbf{B}(\Phi)$. Clearly $\mathbf{B}[\![\phi]\!] \sqsubseteq \mathbf{B}[\![\psi]\!]$ holds iff $\psi \Rightarrow \phi$ is valid, i.e., satisfied by every truth assignment.

Other typical examples include constraint system for streams (the Kahn cs), rational intervals, and first-order theories [27].

2.2 Spatial Constraint Systems

The authors of [16] extended the notion of cs to account for distributed and multi-agent scenarios where agents have their own space for their local information and for 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

$$[c]_i \tag{2}$$

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 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

$$[[c]_j]_i \tag{3}$$

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

$$[c]_i \sqcup [d]_j \tag{4}$$

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 (Spatial Constraint System [16])**.** *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 $[\cdot]_i : Con \to Con$:*

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.

Property S.1 in definition 2 requires space functions to be strict maps (i.e bottom preserving). Intuitively, it states that having an empty local store 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 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*

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

Definition 3 (Global Information). *Let* $\mathbf{C}$ *be an* n*-scs with space functions* $[\cdot]_1, \ldots, [\cdot]_n$. Group-spaces $[\cdot]_G$ *and* global *information* $[\![\cdot]\!]_G$ *of* $G \subseteq \{1, \ldots, n\}$ *are defined as:*

$$[c]_G \stackrel{\text{def}}{=} \bigsqcup_{i \in G} [c]_i \ \ and \ \ [\![c]\!]_G \stackrel{\text{def}}{=} \bigsqcup_{j=0}^{\infty} [c]_G^j \tag{5}$$

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

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.

Kripke Spatial Constraint Systems. We conclude this section with a concrete spatial constraint system from [16]. This constraint system will play a significant role later in Section 4. We basically extend Example 2 by moving from Boolean assignments to *Kripke structures*. Other examples of spatial constraint system for epistemic reasoning are Aumann structures [16].

Definition 4 (Kripke Structures). *An* n*-agent Kripke structure (KS)* M *over a set of atomic propositions* Φ *is a tuple:*

$$M = (S, \pi, \mathcal{R}_1, \ldots, \mathcal{R}_n) \tag{6}$$

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. *The states of 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.

A *pointed KS* is a pair (M,s) where M is a KS and s, called the *actual world*, is a state of M. In the following examples constraints are set of pointed KS. This will allow us to interpret modal formulae as constraints in spatial cs.

Definition 5 (Kripke scs [16]). *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 $Con = \mathcal{P}(\Delta)$, $c_1 \sqsubseteq c_2$ *iff* $c_2 \subseteq c_1$ *and for every* $c \in Con$

$$[c]_i \stackrel{\text{def}}{=} \{(M,s) \in \Delta \mid \forall t : s \xrightarrow{i}_M t \text{ implies } (M,t) \in c\} \tag{7}$$

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 $M \in \mathcal{S}_n(\Phi)$. It is easy to verify that $[true]_i = true$ and $[c_1 \sqcup c_2]_i = [c_1]_i \sqcup [c_2]_i$. Similar to Example 2, a constraint c in $\mathbf{K}(\mathcal{S}_n(\Phi))$ is compact iff $\Delta \setminus c$ is a finite set [16].

A modal language. Modal formulae can be interpreted as constraints in $\mathbf{K}(\mathcal{S}_n(\Phi))$. Such an interpretation will be useful in Section 4.

The modal language $\mathcal{L}_n(\Phi)$ is obtained by extending the grammar for $\mathcal{L}(\Phi)$ in Equation 1 with modalities $\Box_i\phi$ in the standard way.

Definition 6. *Let* Φ *be a set of primitive propositions. The language* $\mathcal{L}_n(\Phi)$ *is given by the following grammar:*

$$\phi, \psi, \ldots \ := \ p \mid \phi \wedge \psi \mid \neg\phi \mid \Box_i\phi \tag{8}$$

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 we interpret each formula ϕ as the constraint $\mathbf{K}[\![\phi]\!] \stackrel{\text{def}}{=} \{(M,s) \in \Delta \mid (M,s) \models \phi\}$ in $\mathbf{K}(\mathcal{S}_n(\Phi))$ where Δ is the set of all pointed Kripke structures (M,s) such that $M \in \mathcal{S}_n(\Phi)$.

Notation 3. *Often, by abuse of notation, we shall suppress the semantic symbols* $\mathbf{K}[\![\ ]\!]$ *from formulae–e.g., we write* $[\phi]_i$ *for the constraint* $[\mathbf{K}[\![\phi]\!]]_i$ *(which is equivalent to* $\mathbf{K}[\![\Box_i(\phi)]\!]$*).*

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 (Boolean Implication). *The constraint systems of the form* $(\mathcal{P}(U), \supseteq)$*, as* $\mathbf{B}(\Phi)$ *in Example 2 and* $\mathbf{K}(\mathcal{S}(\Phi))$ *in Definition 5, are standard examples of Boolean algebras [14]. 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* $\neg c \sqcup d$*, respectively.*

3. Spatial CS's with Extrusion

This is the main section of the paper. We shall introduce our new notion of spatial constraint systems with extrusion (scse) and use it to specify simple examples of mobile and epistemic behaviour. 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 that will be used later on in the Section 4.

3.1 Extrusion as the right inverse of Space

In spatially distributed systems an agent can intentionally 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 for others. The uttered information may be inconsistent with the agent's own beliefs, in particular it could be an intentional 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

$$\text{E.1: } [\uparrow_i c]_i = c. \tag{9}$$

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 7 (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 the 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 3.3.2. In the next section we show that E.1 already allows us to specify meaningful spatial and epistemic behaviour.

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 we mentioned a notion of implication that works for Boolean and Kripke cs. We can use a more general implication by adapting the corresponding notion from Heyting Algebras [30] 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 8 (Heyting Implication and Negation). *Let $\mathbf{C}$ be a constraint system $(Con, \sqsubseteq)$. Define $c \to d$ as*

$$\bigsqcap\{e \mid e \sqcup c \sqsupseteq d\} \tag{10}$$

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

The above construction corresponds to (intuitionistic) implication in lattices that are *frames* [30].

Definition 9 (Frames). *A cs $(Con, \sqsubseteq)$ is said to be a* frame *iff joins distribute over arbitrary meets: More precisely, $c \sqcup \bigsqcap S = \bigsqcap\{c \sqcup e | e \in S\}$ for every $c \in Con$ and $S \subseteq Con$.*

Remark 3. *The previous cs examples in this paper can be all shown to be frames since meets are unions (or intersections) and joins are intersections (or unions) so the distributive requirement is satisfied. Furthermore, if we restrict our attention to cs's of the form $(\mathcal{P}(U), \supseteq)$, as e.g., $\mathbf{B}(\Phi)$ in Example 2 and $\mathbf{K}(\mathcal{S}(\Phi))$ in Definition 5, the operators $\to$ and $\sim$ coincide with the constructions $\Rightarrow$ and $\neg$ defined in Remark 2.*

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

Lemma 1 (Modus-Ponens). *Suppose that the cs $(Con, \sqsubseteq)$ is a frame. Then for every c, d, we have*

$$c \sqcup (c \to d) = c \sqcup d. \tag{11}$$

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

Remark 4. *For the applications examples in this section, we fix an scs $(Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n, \uparrow_1, \ldots, \uparrow_n)$. 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 [29]. 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 (A Hoax). *Suppose that $c \sqcup d = \mathit{false}$. The assertion*

$$[c \sqcup \uparrow_i d]_i \tag{12}$$

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 8 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) followed by the extrusion axiom E.1 (Definition 7) to obtain the following derivation of $[c]_i \sqcup d$.

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

The transformation from Equation 12 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 4 (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*

$$[\sim c \to \uparrow_j d]_j. \tag{13}$$

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

$$[c \sqcup \uparrow_i[\sim c]_j]_i. \tag{14}$$

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 outermost position. The communication should take place if the agents' spaces are placed in parallel. In fact we put together Equations 13 and 14, we derive the expected result.

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

□

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 behaviour of programs.

Example 5 (Mobility). *Let us consider the following assertion:*

$$[e \sqcup \uparrow_i c \to [d]_i]_i. \tag{15}$$

Equation 15 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 1 we can derive $[e \sqcup d]_i$ from $c \sqcup [e \sqcup \uparrow_i c \to [d]_i]$ as follows:

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

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

For a more involved example of extrusion of implication processes consider

$$[e \sqcup \uparrow_i[c \to \uparrow_j[d]_i]_j]_i \tag{16}$$

Intuitively, the implication process $c \to \uparrow_j[d]_i$ is sent from within 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 1 we obtain the following:

$$[e \sqcup \uparrow_i[c \to \uparrow_j[d]_i]_j]_i \sqcup [c]_j \quad \sqsupseteq \quad [e \sqcup d]_i \tag{17}$$

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 *outermost* position in arbitrary nested spaces.

Definition 10 (Global Extrusion). *Let us consider an n-scse $(Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n, \uparrow_1, \ldots, \uparrow_n)$.* Group-extrusion $\uparrow_G$ *and* global extrusion $\Uparrow_G$ *of $G \subseteq \{1, \ldots, n\}$ are defined as:*

$$\uparrow_G c \overset{\text{def}}{=} \bigsqcup_{i \in G} \uparrow_i c \;\; \textit{and} \;\; \Uparrow_G c \overset{\text{def}}{=} \bigsqcup_{j=0}^{\infty} \uparrow_G^j c \tag{18}$$

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

Recall the notion of shared space in Definition 3. The group extrusion $\uparrow_G c$ extrudes c from any space or shared-space of the agents in G. In fact

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

for any $j \in G$.

Global extrusion $\Uparrow_G c$ can pull c into 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 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 6 (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 outermost position if c holds. We can use the global space construction $[\![c \to \Uparrow_G d]\!]_G$ in Definition 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

$$e \sqsupseteq [[\ldots[c]_{i_m} \ldots]_{i_2}]_{i_1} \tag{19}$$

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

$$e \sqcup [\![c \to \Uparrow_G d]\!] \sqsupseteq d. \tag{20}$$

□

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/downwards 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, one can show that the reverse implication is also true: Space continuity implies space completeness.

Proposition 1. *Let $[\cdot]_i$ be a space function of an scs. If $[\cdot]_i$ is continuous then $[\cdot]_i$ is join-complete.*

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 [14].

3.3 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 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) 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 2 by showing something stronger: fibers are closed under finite joins.

Lemma 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.*

The following theorem, an immediate consequence of Lemma 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 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* $[\cdot]_i$*.*

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

3.3.1 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 1 *may* result in $\uparrow_i true \neq true$ for some scs's. 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 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* $c, d \in Con$*.*

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.

3.3.2 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 11 (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 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 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 2 states this impossibility result.

Theorem 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 outline the proof of Theorem 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. One can verify 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 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 2.

Our strategy to prove Theorem 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 (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*

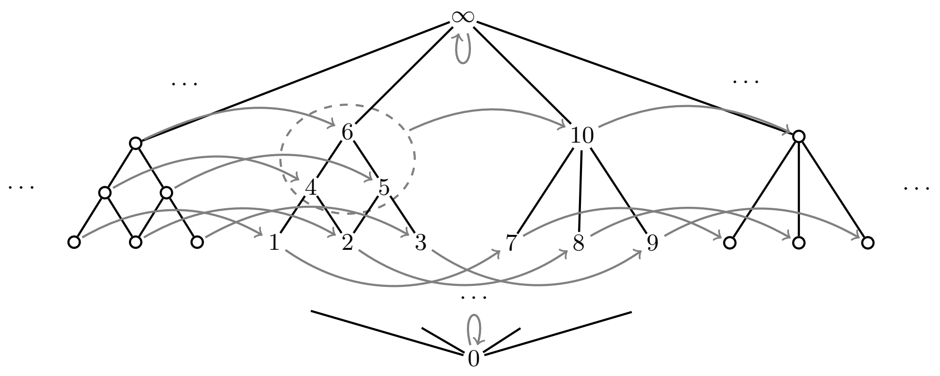

Figure 1: A one-agent scs with a surjective and continuous space depicted by the arrows over $\mathbb{N} \cup \{\infty\}$.

(E.1) $[\uparrow_i c]_i = c$, *(E.2)* $\uparrow_i true = true$ *and (E.3)* $\uparrow_i (c \sqcup d) = \uparrow_i c \sqcup \uparrow_i d$.

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 $c \mapsto \bigsqcap [c]_i^{-1}$.

3.4 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 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.*

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 $[false]_i \neq false$. 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 $[false]_i = false$ reflects the principle, referred to as the *Consistency Axiom* in *belief/doxastic logics*, that no agent can possibly believe the false statement.

Property 1 (Space Consistency)**.** *Let $[\cdot]_i$ be a space function of an scse. Then $[false]_i = false$.*

Nevertheless, for $i \neq j$ we allow the following to occur in an scse: $[c]_i \sqcup [d]_j \neq false$, even when $c \sqcup d = false$. 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. Given $(Con, \sqsubseteq)$ a self-map f over Con is said to be an *order-embedding* iff f preserves and reflects $\sqsubseteq$: i.e, for each $c, d \in Con : c \sqsubseteq d$ 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 S.2 and E.1 one can also show that they reflect $\sqsubseteq$. Thus, extrusion functions are order-embeddings:

Property 2 (Extrusion Embedding)**.** *Let $\uparrow_i$ be a distributed extrusion function of an scse. Then $\uparrow_i$ is an order-embedding.*

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 cs space functions necessarily preserve distinctness, i.e., they are *injective*. In particular,

Proposition 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.*

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

A noteworthy corollary of Property 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 [15].

Corollary 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 meet-complete.*

Notice that from Proposition 3, Corollary 2, and Property 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 1). One may then wonder if space functions from scse's are already continuous. A negative answer is given in the example below.

Example 7 (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 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 $[\bigsqcup S]_1 = [(1,0)]_1 = (0,1) \neq (0,0) = \bigsqcup\{[(0,n)]_1 \mid n \geq 0\}$.* □

The above example also shows an application of Theorem 3 to derive an extrusion function for a rather simple scs. Notice that we could not have applied Theorem 1 because the $[\cdot]_1$ was shown not to be continuous. In the Application section we will derive extrusion functions for a meaningful and more involved scs using Theorem 1.

Galois Connections. We conclude this section by stating a pleasant correspondence between space and extrusion. In Example 7 we used Theorem 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 of (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.

Property 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 $c \in Con$.*

It follows from Property 4 that the pair $(\uparrow_1, [\cdot]_1)$ in Example 7 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 4.

Example 8. *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 4 we can conclude that $(\uparrow_i, [\cdot]_i)$ is not a Galois connection. (One can also verify using Property 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

$$[c]_i \sqsubseteq d \Leftrightarrow c \sqsubseteq \uparrow_i d \tag{21}$$

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

4. Applications: Kripke scs with extrusion & Belief with utterance

In Section 3.3 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 5) as an application of the results we obtained in Sections 3.3 and 3.4.

Recall that we can associate a modal language (Definition 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 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 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

$$[c]_i \stackrel{\texttt{def}}{=} \{(M, s) \in \Delta \mid \forall t : \text{ if } s \xrightarrow{i}_M t \text{ then } (M, t) \in c\} \tag{22}$$

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

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.

Remark 5. *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 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 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 22 we conclude that $(M', s') \in [\mathit{false}]_i$ thus violating space consistency. Property 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 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 (left-total space-consistency). *$\mathcal{M}^{lt}$ satisfies space consistency.*

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 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 22 we conclude that if $(M, s) \in c$ then $(M, s') \in c$. Thus, surjectivity is not satisfied by $[\cdot]_i$ since for every $d \in \mathcal{P}(\mathcal{M}^{\mathrm{lt}})$, $[d]_i \neq \{(M, s)\}$. Thus $\mathbf{K}(\mathcal{M}^{\mathrm{lt}})$ 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 $\mathcal{M}^{\mathrm{lt}}$ 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 $\mathcal{M}^{\mathrm{ltu}}$ 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 $M \in \mathcal{M}^{\mathrm{ltu}}$.

Proposition 5 (left-unique surjectivity)**.** *The set $\mathcal{M}^{ltu}$ satisfies surjectivity.*

We now have an scs $\mathcal{M}^{\mathrm{ltu}}$ 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 3.3 with Theorems 1 and 3.

We cannot apply Theorem 3 because the spatial functions of $\mathcal{M}^{\mathrm{ltu}}$ 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 $\mathbf{K}(\mathcal{M}^{\mathrm{ltu}})$ 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 6. *The space functions of $\mathbf{K}(\mathcal{M}^{ltu})$ are continuous.*

Therefore we can apply Theorem 1 and derive the following extrusion function for each $[\cdot]_i$ in $\mathbf{K}(\mathcal{M}^{\mathrm{ltu}})$:

$$\uparrow_i : c \mapsto \bigsqcup [c]_i^{-1}. \tag{23}$$

Furthermore, we can show that the construction in Equation 23 is equivalent to the intensional definition given below.

Lemma 3. *Let $\uparrow_i$ be defined as in Equation 23 over the Kripke scs $\mathbf{K}(\mathcal{M}^{ltu})$. Then*

$$\uparrow_i(c) = \{(M, t) \in \Delta \mid \exists s :\ s \xrightarrow{i}_M t \text{ and } (M, s) \in c\} \tag{24}$$

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

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

Definition 12 (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 23.*

It follows from Property 4 that in the derived scse, space and extrusion form a *Galois connection*.

Corollary 3. *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 6 with modalities to express utterances. The intended meaning and properties of the formulae in the extended language will be given from the scse $\mathbf{K}(\mathcal{M}^{\mathrm{ltu}})$ we derived in the previous section (Definition 12). 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 5 and extending it with modalities U_i.

Definition 13 (Syntax of the BU_n logic)**.** *Let $\mathcal{L}_n^{BU}(\Phi)$ with $n \geq 1$ be the language built from a set of primitive propositions Φ by the following syntax:*

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

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

Before giving semantics to BU_n, it is convenient to give a dual spatial/epistemic intuition about its formulae: The *belief* modality $B_i\varphi$ holds true in the world s (of a KS M) iff φ is true in every world t that i considers possible from s, i.e., in every $t \in \mathcal{W}_i(M, s)$. We can think of the t's as worlds of agent i. Thus $B_i\varphi$ means that agent i *believes* φ to be true (in all his possible worlds). The *utterance* modality $U_i\varphi$ holds true in t iff t is a world of agent i and φ holds in the (outside) world s that t comes from (i.e., the world s such that $t \in \mathcal{W}_i(M, s)$). Thus $U_i\varphi$ implies that φ holds true in the world s whenever the utterance is stated true in an (inside) world t of agent i that comes from s (i.e. a world t such that $t \in \mathcal{W}_i(M, s)$).

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

$$B_iU_i\varphi \Leftrightarrow \varphi. \tag{25}$$

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

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

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:

$$\mathcal{O}_i(\varphi) \Leftrightarrow (B_i\varphi) \wedge \varphi. \tag{26}$$

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

$$\mathcal{H}_i(\varphi) \Leftrightarrow (B_i\neg\varphi) \wedge \varphi \tag{27}$$

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

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

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:

$$B_i\varphi \Rightarrow \hat{B}_i\varphi \text{ and } U_i\varphi \Rightarrow \hat{U}_i\varphi \tag{28}$$

The formulae in Equation 28 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 $\mathbf{K}(\mathcal{M}^{\mathrm{ltu}})$. Recall our definition of the negation constraint $\sim c$ (Definition 8) and that $\mathbf{K}(\mathcal{M}^{\mathrm{ltu}})$ is also a frame (Remark 3).

Definition 14. *Let* $\mathbf{K}^{\uparrow}(\mathcal{M}^{ltu}) = (Con, \sqsubseteq, [\cdot]_1, \ldots, [\cdot]_n, \uparrow_1, \ldots, \uparrow_n)$ *be the scse in Definition 12. Given* φ *in* $\mathcal{L}_n^{BU}(\Phi)$*, its denotation* $\mathbf{K}^{\uparrow}[\![\varphi]\!]$ *is inductively defined as follows.*

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

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 behaviour of utterance, opinion, and hoaxes in Equations 25, 26, 27 and 28.

Proposition 7. *The formulae in Equations 25, 26, 27 and 28 are valid in* BU_n.

Notice that $\varphi \Rightarrow \psi$ is valid in BU_n iff $\mathbf{K}^{\uparrow}[\![\psi]\!] \sqsubseteq \mathbf{K}^{\uparrow}[\![\varphi]\!]$. It follows from Corollary 3 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. $\varphi \Rightarrow B_i\psi$ *is valid in* BU_n *iff* $U_i\varphi \Rightarrow \psi$ *is valid in* BU_n.

Intuitively Corollary 4 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.

5. Concluding Remarks and Related Work

We introduced the notion of scse as complete algebraic lattices with self-maps representing space and extrusion. The central design concept behind scse's is the view of extrusion as a right inverse of space, and utterance as a right inverse of belief. We used scse's to specify examples of distributed and epistemic behaviours such as spatial mobility, belief, utterance, lies and opinions. We developed scse's by building upon notions and concepts from order (domain) theory, epistemic (doxastic) theories, and the algebraic treatment of logic in [10, 30]. We investigated properties relating space and extrusion such as consistency, automorphisms and Galois connections. We studied the problem of finding, for given space functions, the corresponding extrusion functions. We illustrated an application of our results by deriving extrusion functions for an existing spatial constraint system and then use the resulting scse to give semantics to a simple modal logic of belief and utterance. In this logic, the modalities for belief and utterance were shown to form a Galois connection.

Our scse's can be used as constraint systems for concurrent constraint programming (ccp) calculi. This way processes in these calculi would be able to express spatial mobility and epistemic/social behaviours. The issue of extending ccp calculi to provide for distributed information has been previously addressed in [24]. In [11, 24] processes can send constraints using communication channels much like in the π-calculus. In [19] temporal ccp process can transmit variables using existential and universal quantification. More recently, in [20] the authors added the notion of link mobility to spatial and linear ccp using a proof-theoretical approach. Our approach differs from these works in both conception and technical development. We view extrusion/utterance as inverses of space/belief and develop this concept using order-theory and epistemic logic.

Epistemic logics have been widely applied to distributed systems; [13] 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 [25] the logical properties and consequences of introducing inverse modalities in a generic modal logic is thoroughly explained. Also in [22] the authors put forward an extension of Hennessy-Milner logic with a reverse modality for expressing concurrent behaviour. In this paper, 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. In future work we plan to develop an algebraic presentation of scse's by building upon the axiomatisation of logics with reverse modalities given in [25].

Social phenomena such as lies, utterance, opinions have been recently studied in epistemic (doxastic) logic [26, 28, 28, 29]. We follow [29] 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 [26, 28, 29] 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 [26, 28, 29] 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 [9], a representative process calculus for spatial mobility. 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 charaterize 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 [7, 8]. 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. In effect, the concept of freshness of a name is exploited to control the flow of information. It would be interesting to see how a name quantified scs would look and to study the relationship with the framework in [7, 8].

Finally, the process calculi in [3, 6, 12] 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.

Acknowledgements. We would like to thank Moreno Falaschi, Michell Guzman, Sophia Knight and Carlos Olarte for helping to improve the presentation of this work.

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

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

[6] 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.

[7] 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.

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

[9] 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.

[10] 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.

[11] 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.

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

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

[14] 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.

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

[16] 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.

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

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

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

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

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

[26] 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.

[27] 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.

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

[29] 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.

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