Naval Research Laboratory
Washington, DC 20375-5320

NRL/5540/MR—2023/2

Basic Distributed Logic for Spiders

DR. GERARD ALLWEIN

CHRISTOPHER BELMONTE

Center for High Assurance Computer Systems Branch
Information Technology Division

August 17, 2023

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
17-08-2023	NRL Memorandum Report	10-01-2023 – 09-30-2023

4. TITLE AND SUBTITLE

Basic Distributed Logic for Spiders

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
062235N

6. AUTHOR(S)

Dr. Gerard Allwein and Christopher Belmonte

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
6C59

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
4555 Overlook Avenue, SW
Washington, DC 20375-5320

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL/5540/MR--2023/2

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
4555 Overlook Avenue, SW
Washington, DC 20375-5320

10. SPONSOR / MONITOR'S ACRONYM(S)
NRL

11. SPONSOR / MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A: Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This is a report on basic Distributed Logic.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
					Dr. Gerard Allwein
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	U	35	**19b. TELEPHONE NUMBER** *(include area code)*
U	U	U			(202) 404-3748

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

This page intentionally left blank.

CONTENTS

This page intentionally left blank

EXECUTIVE SUMMARY

This is a report on using Distributed Logic for spiders. A spider is a realtime monitor for FPGA applications. Each spider is specific to an application. The mental picture is that the body of a spider sits over an application and its legs stick down into the application. Signals travel up and down the legs. The signals to the spider are *recognition* signals and report on conditions in the application. Signals from the spider are *mitigation* signals and direct the application to perform behavior outside of its normal operation. The mitigation may require specialized code in the application to respond properly to the mitigation. A spider is written in any language for FPGA applications such as VHDL or Verilog. We prefer to use ReWire, which is a functional language that compiles to VHDL or Verilog. A spider in compiled form is then included along with the application and the lot is further processed by vendor tools to eventually become a file loaded onto an FPGA. This file is the FPGA application + spider and defines that application + spider. An FPGA can be thought of a clean slate of transistors what only receive their instructions on how to act by virtue of this file.

Distributed Logic was developed by the PI expressly for distributed systems such as FPGA applications. An FPGA application is similar to a circuit board shrunk down to fit on a chip. Like a circuit board, an application is a collection of components all operating in parallel and exchanging signals. The usual logics tend to get a bit out of hand in such an environment because they have no facilities for directly representing this distributed structure. Instead, they rely upon layers of encoding, which makes them clumsy and prone to misuse. Distributed Logic incorporates a graph of nodes and arcs directly as part of the logic; no other logic does this. The graph allows statements in Distributed Logic to mirror the structure of the application. Each component gets its own local logic and distributed operators connect the local logic. Each local logic is a generally a modal logic so that it can state properties of components' state machines. The distributed operators are also modal operators but take formulas in one local logic and construct formulas in another local logic.

This notion of distribution is much wider than FPGA applications. We have shown in other research how to distribute relation algebras and other logics. In general, this distribution appears to be *orthogonal* to many logic constructs in that it usually does not affect them and hence it has a wide applicability.

This page intentionally left blank

BASIC DISTRIBUTED LOGIC FOR SPIDERS

1. INTRODUCTION

A *spider* is a form of realtime monitor for checking logical statements about an FPGA application and for mitigating dangerous behavior when it is recognized as failing the logical statements. We also use the term *spider* as verb, i.e., to *spider* an application, which means to add spiders to an application to check its behavior during runtime. It is possible to have hierarchies of spiders with the upper level spiders checking lower level spiders. Ultimately, we wish to compile the logical statements into FPGA code that is further compiled along with an application to produce the bitstream that is loaded onto an FPGA.

The logical statements are written in Distributed Logic (DL), which the PI has developed expressly for distributed systems. Each distributed system has its own notions of high assurance. Many of these can be coded in DL. One way to view a distributed system for an FPGA application is as a graph with nodes for components and arcs for connecting the components. In DL, each component gets its own local logic. This local logic is typically a modal logic over a state machine that defines the component's behavior. Two components have states that can co-occur with each other in parallel. DL semantics represents this as a relation corresponding to the arc in the graph. To make that arc part of logic, DL includes distributed connectives that are underwritten or made true by virtue of the distributed relations in much the same way as a modal operator for a local logic is underwritten by a next-state relation; the mathematics is the same even though underlying relations have quite a different character.

DL is not tied to any one kind of distributed system. Logical properties are simply properties. The main feature DL offers is the ability to lift the distribution structure directly from the distributed system homomorphically into the logic. This makes tying statements in DL to the system much more straightforward than using other logical systems. The modal operators within a local logic can be made quite general; they can have strong or weak properties and be n-ary in their inputs. The same holds true for the distributed operators. The distributed relations in the semantics underlying these distributed operators can be any relation that makes sense for the specific use of DL. In addition, there is some work being done on mechanizing DL using the Coq theorem prover.

The concept of a spider is not particularly tied to FPGAs in the sense that if the recognition part is expressed in DL, the spider could be a monitor for any distributed system as long as there is an implementation of what it means to monitor a system. Given an implementation, it becomes possible to think about mitigation in terms of the same kind of implementation.

2. BASIC CLASSICAL PROPOSITIONAL LOGIC

Given the variety of symbols used in this paper, the following table may be of some help. Distributed Logic uses a graph of localities. A locality is a combination of a topological space and a logic or algebra interpreted in the space. The term *local* in the table below is used to restrict a symbol to a single locality.

Manuscript approved August 15, 2023.

Variables

Symbol	Use	Typeface
P, Q, etc.	propositional variables	upper case italic, center of the alphabet
x, y, etc.	points of a space	lower case italic, end of the alphabet
S	set of states	upper case italic
H,	set of points of a topological space	upper case italic, center of the alphabet
$\mathcal{H}$, $\mathcal{K}$	local binary relations	
math cal.		
$\mathcal{F}$, $\mathcal{G}$, $\mathcal{R}$, $\mathcal{S}$	distributed binary relations	math cal.
h, k, l	localities or graph nodes	math sans serif
a, b, c, etc.	syntactic values	lower case italic, beginning of alphabet
v	model theoretic value	lower case italic
state	state variable of a finite state machine	sans serif
p, q	FPGA application variable	math italic
s	simple function	math italic

Operators

Symbol	Use	Typeface
$\wedge$	conjunction or lattice meet	math
$\vee$	disjunction or lattice join	math
$\neg$	negation	
$\forall$, $\exists$	universal and existential quantifiers	
$\mathcal{P}$	power set operator	math

Table 2.1: Symbols

2.1 Syntax

We need a language, preferably a logical language, in order to talk about the meaning or semantics of DL. This language is known as a *meta-language*. A semantics is a map from the object language, in our case DL, to a mathematical model consisting of sets of states, functions, and relations. The semantics is expressed in the meta-language. The semantics is generally given in a *first-order language*. This is a language that allows us the usual logical notions (negation, conjunction, etc.) but also allows us to quantify over individuals of some domain, say a domain of states or a domain of signals. The following first 4 symbols are used in both DL and the meta-language. The two quantifier symbols, $\forall$ and $\exists$ are used in the meta-language.

Symbol	Logical Meaning	Colloquial Example
$\neg$	Negation	Not
$\wedge$	Conjunction	And
$\vee$	Disjunction	Or
$\supset$	Implication	If-then
$\forall$	Universal Quantifier	For Every (Each)
$\exists$	Existential Quantifier	There Exists at Least One
$\in$	"In" Relation	Element of a set, e.g., $x \in P$
		"the element x is in the set P"
$\subseteq$	"Subset Relation"	Subset and Possibly Equal

Table 2.2: The Usual Symbols of Logic and Set Theory and Their Meanings

The formula $P \supset Q$ can be read as either "if P, then Q" or as, more commonly, "P implies Q". Set theory is itself written in a first-order language. Set theory has a set of axioms and a set of rules telling us how to get new statements in set theory from old statements.

We frequently use the locution "P iff Q" for

$$(P \text{ implies } Q) \text{ and } (Q \text{ implies } P).$$

We will not axiomatize classical propositional logic here but note that one can interpret the formulas as sets in a Boolean lattice of sets and use the algebraic Boolean lattice axioms.

2.2 Evaluating Logic Formulas

A logic, any logic, is expressed in a language that we call the *object language*. The *interpretation* of the logic is a map from formulas of the logic to (usually) mathematical terms. This parallels non-formal languages such as English. Think of the syntax of English as given by those diagrams you may have learned in grade school for diagramming sentences. The semantics, or meaning, is usually given in English itself but that is via necessity. If we wanted, we could interpret the language in another language, say, German. The two languages are not considered on par with each other when the second is used in semantics. The language we use to express the meaning of formulas in the object language is called the *metalanguage*. We generally use the language of mathematics for interpreting the logical language. In our case, we use points (representing states), sets, and relations. Functions may be used but these are just binary relations (two-place) relations with special properties, i.e., they are defined over their entire domain, and that no single element can be mapped to two distinct elements in the codomain.

We will interpret a logic in terms of *points*, which you can think of as states in a computer system. For us, they will typically arise with respect to components in an FPGA application. The term *state* can mean

different things to different communities. We prefer "point", which is more generic but move back and forth between the two depending upon how generic we are expressing a property. A state represents a cross-section through a system where the system is thought of as having a set of possible trajectories though a state space. The allowable trajectories give a notion of what states can follow other states, the entire space is a sea of states. All of the spaces we use to evaluate logic formulas will at least form Stone spaces (see Section 3.2).

A proposition in a logical language is some linguistic formula that we wish to reason with, say $x = 2$ where 2 is considered fixed and x is considered to be variable. As a linguistic entity, it is just that, i.e., a string of symbols with some internal structure but that structure is not identifying any specific value for x. The proposition $x = 2$ is uninterpreted at this point in the presentation. We can consider the collection of points that make this proposition true. Which points are those? All the points in which the variable (memory location) x actually has the same value of the interpretation of the constant 2. We expect any interpretation is sensible in that it evaluates the constant 2 as the number 2. The constant 2 is a linguistic entity and nothing more. The number 2 is an abstract mathematical object. You cannot hold a mathematic object in your hand like you can cut out the 2 on a piece of paper and hold it.

We frequently collect together all the points that turn a proposition true and use that collection and the proposition interchangeably with context disambiguating whether we mean the linguistic entity or the set of points making that proposition true.

A collection of points is considered a proposition in a logic of sets. One way to present this is in terms of a satisfaction relation $\models$. The locution $x \models P$ is read as "the object (say a state) x *satisfies* or makes true the proposition P". In symbols, the semantics $[\![P]\!]$ of a proposition P is specified as

$$x \in [\![P]\!] \text{ iff } x \models P.$$

Typically, we elide the brackets and simply use $x \in P$, and, by virtue of the iff, move freely back and forth between thinking of P as a set and P as a linguistic proposition. The reason we are allowed to do this is that we can define

$$P \equiv Q \stackrel{def}{=} (P \supset Q) \wedge (Q \supset P),$$

then $\equiv$ defines an equivalence relation (reflexive, symmetric, and transitive) given the axioms and rules of classical logic. As such, it divides the collection of logical formulas into equivalence classes which we denote $[P]$ for a proposition P. Hence

$$[P] = \{Q \mid P \equiv Q\},$$

so that

$$[P] = [Q] \text{ iff } P \equiv Q.$$

Algebraically, we are dividing the set of formulas out by bi-implication ($\equiv$). Given the semantic rules, we have

$$[P] = [Q] \text{ iff } \forall x(x \models P \text{ iff } x \models Q).$$

In terms of sets with $\in$ for $\models$, we have

$$[P] = [Q] \text{ iff } \forall x(x \in P \text{ iff } x \in Q).$$

A proposition P is then considered a statement about states. A typical proposition is

$$P(x) = \text{The state is } x,$$

where x is intended to range over a set of points. Notice that x represents a "hole" in the statement. This could equally have been stated as

$$P(-) = \text{The state is } - .$$

As there is only a single hole, this is adequate. To represent a more complicated proposition, say, $Q(-,-)$, we would have to name the holes; this is precisely what variables do. Also, the variable "x" in $P(x)$ is determined, i.e., given a value, in the surrounding mathematical environment, much like for a function f, say,

$$f(x) = x^2$$

is a formula where the value of x is set by the surrounding mathematical environment. We are used to treating x as some undefined quantity ranging over some set, in this case, the set of real numbers. The notation $P(x)$ is entirely analogous. At the level of language, f and x are linguistic entities. We frequently think of them as already interpreted as a function and a point. Similarly, we can treat P and x either as linguistic entities or as a set and a point. Semantically, we can also think of P as a propositional function, i.e., a function from the set of points to the set of values $\{\top, \bot\}$ where $\top$ represents *true* and $\bot$ represents *false*, and which we shorten to merely "proposition".

Now we will connect the logical language with the model theoretic setting where the locution "model theoretic" means a collection of situations where formulas are either true or false.

Using a concrete situation such as a finite state machine, to say that x is a state, we frequently rely upon set theoretic notation, i.e.,

$$x \in \{s_0, s_1, s_2, s_3\}.$$

Also, since there is no order in sets and repetitions do not matter,

$$\{s_0, s_1, s_2, s_3\} = \{s_2, s_0, s_1, s_2, s_3\}.$$

where the = sign is standing for equality in the meta-language.

Propositions are either *true* or *false* in a particular setting. The setting for our concrete situations are finite state machines and we must specify a particular point to get a completely defined setting. So we are left with specifying the truth or falsity of a proposition at each point.

We can treat P as a linguistic entity in a logical language or we can treat it as set of those points at which the proposition is true.

To say that $P(x)$ is true only for $x = s_1$ can be represented as

$$P = \{s_1\}.$$

Hence

$$P(x) \text{ is true iff } x \in \{s_1\}.$$

The short hand "iff" means this statement stands for

$$(P(x) \text{ is true }) \text{ implies } (x \in \{s_1\}) \quad \text{and} \quad (x \in \{s_1\}) \text{ implies } (P(x) \text{ is true }).$$

In effect, the iff locution allows us to use $P(x)$ interchangeably with the set $\{s_1\}$. We often elide the parentheses since "iff "and "implies" only make sense between complete statements. x by itself is not a complete statement whereas $x \in P$ is a complete statement. To see this, just convert it into English, e.g., x is in P, where as x or P by itself is not.

It is helpful to switch between P as a linguistic entity in a language and under an interpretation as a set. The latter is to treat P as a set of points. A proposition as a set of points is called a *UCLA proposition*[1]. To evaluate logic formulas, we use the symbol $\models$. The term *models* ($\models$ is its symbol) is a relation between points and propositions and hence a point-proposition pair is either in the relation it is not. In other words, the point-proposition is considered true under the relation or it is not. Similarly, $\in$ when used in $a \in P$ is a relation between points and P, this latter as an UCLA proposition.

Definition 2.2.1 A Boolean set lattice $BA(\mathsf{h}) = \langle BA(H), \cap, \cup, \neg \rangle$ is a collection of sets of points or states $BA(H)$ taken from a set of points H, and the set operations: set intersection $\cap$, set union $\cup$, and set complement $\neg$. The top of the lattice, $\top^{\mathsf{h}}$, is the entire set of points H and the bottom of the lattice, $\bot^{\mathsf{h}}$, is the empty set $\emptyset$, i.e., none of the points. □

We will use the locution $P \in \mathsf{h}$ to mean that $P \in BA(H)$.

Definition 2.2.2 A *local classical frame* or *classical locality*, shortened in this paper to just *locality*, is a structure $H = \langle H, BA(\mathsf{h}) \rangle$ such that H is a collection of points called, generically, a *domain*. We use the same symbol for the frame and its collection of states, and let use disambiguate meaning. $BA(\mathsf{h})$ can be alternately thought of as (1) a classical propositional logic, or (2) as a collection of *neighborhoods* or sets of states that are subsets of H and the entire collection is closed under the Boolean operations. Hence $BA(\mathsf{h})$, in this latter case, is a Boolean set algebra. The top proposition is $\top^{\mathsf{h}}$ thought of as always *true* and the bottom proposition $\bot^{\mathsf{h}}$ is thought of as always false. As sets in the set algebra, the set $\top^{\mathsf{h}}$ is the top of the Boolean lattice of sets and is the collection of all states, i.e., H. The set $\bot^{\mathsf{h}}$ is the bottom and is the empty set $\emptyset$. □

A diagram of a locality that includes a local logic and its interpreting states is

[1] The University of Southern California, logicians there used the notion.

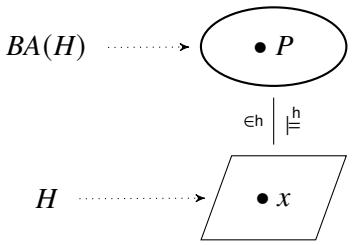

Figure 2.1: Intuitive Diagram of a Locality

where $\models^{\mathsf{h}}$ is used if $BA(H)$ is a collection of propositions and $\in_{\mathsf{h}}$ is used if $BA(H)$ is a collection of sets. The h in $\in_{\mathsf{h}}$ indicates the frame h and is the same as the unadorned $\in$ of bog standard set theory.

Since the set $BA(H)$ is either a set of formulas or a collection of sets of states, we will use the locution $P \in BA(H)$ to mean either with use disambiguating meaning. Given the fluidity of how to consider a proposition P, it usually makes no difference which you choose.

The symbols

$$x \models^{\mathsf{h}} P \qquad x \not\models^{\mathsf{h}} P$$

mean (respectively) the proposition P is true of point x at locality h and the proposition P is false of point x at locality h. The former is an assertion that P is true of x at locality h and the latter is an assertion that P is false of x at locality h. This notation frees us to place extra restrictions on x and on $\models$. In fact, you already have seen one where $\models^{\mathsf{h}}$ is restricting the $\models$ relation to the locality h.

This notation allows us to treat the *models* symbol, $\models^{\mathsf{h}}$, as the set theoretic membership $\in$ at locality h. The latter can also be displayed as $\in_{\mathsf{h}}$. An *atomic* proposition P from a logic has no internal logical structure, i.e., no embedded $\neg$, $\wedge$, etc.. An atomic UCLA proposition is one where there are no other propositions Q such that $\emptyset \subseteq Q \subseteq P$ (where the empty set $\emptyset$ stands for falsity $\perp$).

Let Aprops be the set of *atomic propositions*. A valuation $[\![-]\!]$ turns a proposition into a propositional function. The $\models^{\mathsf{h}}$ relation is extended inductively for $[\![-]\!]$ a valuation on the atomic propositions:

- $x \models^{\mathsf{h}} P$ iff $x \in_{\mathsf{h}} [\![P]\!]$ for $P \in$ Aprops,
- $x \models^{\mathsf{h}} \neg P$ iff $x \not\models^{\mathsf{h}} P$, i.e, $x \notin_{\mathsf{h}} [\![P]\!]$,
- $x \models^{\mathsf{h}} P \wedge Q$ iff $x \models^{\mathsf{h}} P$ and $x \models^{\mathsf{h}} Q$,
- $x \models^{\mathsf{h}} P \vee Q$ iff $x \models^{\mathsf{h}} P$ or $x \models^{\mathsf{h}} Q$,
- $x \models^{\mathsf{h}} P \supset Q$ iff $x \not\models^{\mathsf{h}} P$ or $x \models^{\mathsf{h}} Q$,

Incidentally, the clause for $P \supset Q$ can be written

$$a \models^{\mathsf{h}} P \supset Q \text{ iff } a \models^{\mathsf{h}} P \text{ implies } a \models^{\mathsf{h}} Q.$$

In general, these are the clauses for evaluating classical logic. The logic of the meta-language is also classical and hence when reasoning in it, we use similar valuations.

Notice that we can also code these in terms of set theory. Let H be the collection of all points and here, $\sigma : \text{Aprops} \to BA(\mathsf{h})$ is a valuation on the atomic propositions that returns a collection of points where for any $P \in \text{Aprops}$, $\sigma P \in BA(\mathsf{h})$:

- $a \models^{\mathsf{h}} P$ iff $x \in_{\mathsf{h}} \sigma P$ — for P an atomic proposition,
- $a \models^{\mathsf{h}} \neg P$ iff $x \in_{\mathsf{h}} H - P$ iff $x \notin_{\mathsf{h}} P$ — set difference,
- $a \models^{\mathsf{h}} P \wedge Q$ iff $a \in_{\mathsf{h}} P \cap Q$ — set intersection,
- $a \models^{\mathsf{h}} P \vee Q$ iff $a \in_{\mathsf{h}} P \cup Q$ — set union,
- $a \models^{\mathsf{h}} P \supset Q$ iff $a \in_{\mathsf{h}} \neg P \cup Q$ — set inclusion, i.e., if for all a, $a \models^{\mathsf{h}} P \supset Q$, then $P \subseteq Q$.

This means that we can treat the collection of all propositions aa a Boolean set lattice.

Next, we extend the *models* symbol, $\models^{\mathsf{h}}$, as a relation, to do more work by leaving the left hand side unspecified:

- $\models^{\mathsf{h}} P$ iff for all $a \in_{\mathsf{h}} H$, it is the case that $a \models^{\mathsf{h}} P$. Equivalently, $\models^{\mathsf{h}} P$ if and only if for all $a \in_{\mathsf{h}} \top^{\mathsf{h}}$ (where $\top^{\mathsf{h}}$ is the top of the Boolean set algebra $BA(\mathsf{h})$, i..e. H here), it is the case that $a \models^{\mathsf{h}} P$.
- $\not\models^{\mathsf{h}} P$ iff there is some $a \in_{\mathsf{h}} H$ and $a \not\models^{\mathsf{h}} P$.
- A local logic at h is consistent just when for all $a \in_{\mathsf{h}} H$, it is the case that for all propositions P, not both $a \models^{\mathsf{h}} P$ and $a \not\models^{\mathsf{h}} P$.

This semantics is part of the meta-language, which is the language we use to talk about the language of the logic in which we are interested. Typically, we are not so pedantic and simply use variables $x, y, \ldots$ in both the meta-language and the logic letting use disambiguate what is meant. We frequently must quantify over the meta-language to get the correct level of generality. In that case, it is possible to formalize the meta-language a bit with, in addition to the clauses above:

- $a \models^{\mathsf{h}} \forall x(P(x))$ iff for all $v, a[v/x] \models^{\mathsf{h}} P(x)$,
- $a \models^{\mathsf{h}} \exists x(P(x))$ iff there is some $v, a[v/x] \models^{\mathsf{h}} P(x)$.

where the variable v is a quantified variable in the meta-language, a is an arbitrary point in the meta-language, and x is a variable in the object or logical language in which we are interested.

So a point a becomes not merely a list of values for all the propositions but also records values for individual variables in the logic. That is, $a[v/x]$ means that when is evaluating $P(x)$, it looks up the value of x in its list of variable-value pairs and uses the value stored there, call it v.

3. NORMAL DISTRIBUTED LOGIC

The term "normal" with respect to classical modal logics has a technical meaning: the modal operator distributes across conjunctions and takes *true* to *true*. Let conjunctions be denoted by $\wedge$ and *true* by $\top$. Hence

$$\Box(P \wedge Q) \equiv \Box P \wedge \Box Q, \quad \Box \top = \top$$

for the modal necessity operator $\Box$. The formula $\Box P$ says that P is necessarily true. You can think of this as being true in all possible worlds accessible from the current world. A world is collection of points, and accessible means a binary relation tell you how to get from one world to another.

The other usual modality is possibility, e.g., $\Diamond P$. This has the interpretation the P is true in at least one world accessible from the current world.

3.1 Normal Distributed Modal Logic

DL assumes a graph of localities for which we typically use h and k to indicate localities. These are the worlds of modal logic. Whereas modal logic does not have the concept of world in the logic (only in the models), DL has the concept of world in the guise of a graph of worlds (localities). The distributed modalities take a proposition at one locality into a proposition an another. A local modality takes propositions from one locality to the same locality. So a local modality is distributed with but a single locality.

We use the connective $[f^{\circ}\rangle$ is a distributed possibility, and $[h^{\Box}\rangle$, $[k^{\Box}\rangle$ as local necessities. We denote local arrows in the graph using the unadorned math italic h, k, etc.. We denote distributed arrows using the unadorned f and g. Hence an arc $h : \mathsf{h} \rightharpoonup \mathsf{h}$ represents an endo-arrow. An arc $f : \mathsf{h} \rightharpoonup \mathsf{k}$ represents a distributed arrow from h to k. We will generally assume an endo-arrow for any node h, k, etc. Out of practicality, more could be added depending upon the application. We will always type f as $f : \mathsf{h} \rightharpoonup \mathsf{k}$, however the type of g will vary as needed.

There are no restrictions on a DL's graph except that it be finite. Semantically, the arcs of the graph specify accessibility relations among the localities rather than modal connectives. The reason is that a single distributed relation can support several different modal connectives (see [1] but generalized to be distributed connectives as in [2]).

3.2 Frames and Models

The interpretation of formulas follows the usual modal logic interpretations with care for the distributed structure. Let $f : \mathsf{h} \rightharpoonup \mathsf{k}$, the evaluation conditions for $[f^{\Box}\rangle$ and $[f^{\circ}\rangle$ are

$$x \models^{\mathsf{h}} [f^{\Box}\rangle Q \text{ iff } \forall y \in K(\mathcal{F}xy \text{ implies } y \models^{\mathsf{k}} Q), \text{and}$$
$$x \models^{\mathsf{h}} [f^{\circ}\rangle Q \text{ iff } \exists y \in K(\mathcal{F}xy \text{ and } y \models^{\mathsf{k}} Q).$$

The local connectives are interpreted similarly for local relations called *endo-relations*, say $\mathcal{H}$ and $\mathcal{K}$, for $\mathcal{F}$. As is typical, the set algebra provides a $\models$ relation (respecting localities) using the set theoretical membership relation, $\in$. Hence this formula is evaluated thusly

$$x \in_{\mathsf{h}} [f^{\Box}\rangle Q \text{ iff } \forall y \in K(\mathcal{F}xy \text{ implies } y \in_{\mathsf{k}} Q), \text{and}$$
$$x \in_{\mathsf{h}} [f^{\circ}\rangle Q \text{ iff } \exists y \in K(\mathcal{F}xy \text{ and } y \in_{\mathsf{k}} Q).$$

where Q is now a set of points in $BA(\mathsf{k})$. The proposition Q, when viewed as a set of points, is a termed a UCLA proposition.

The backward connectives, $[f^{\Box}\cdot\rangle$ and $[f^{\circ}\cdot\rangle$ have the following valuation conditions:

$$y \models^{\mathsf{k}} [f^{\Box}\cdot\rangle P \text{ iff } \forall x \in H(\mathcal{F}xy \text{ implies } x \models^{\mathsf{h}} P), \text{and}$$
$$y \models^{\mathsf{k}} [f^{\circ}\cdot\rangle P \text{ iff } \exists x \in H(\mathcal{F}xy \text{ and } x \models^{\mathsf{h}} P).$$

In the set algebras, these formulas is evaluated thusly

$$y \in_{\mathsf{k}} [f^{\Box}\cdot\rangle P \text{ iff } \forall x \in H(\mathcal{F}xy \text{ implies } x \in_{\mathsf{h}} P), \text{and}$$
$$y \in_{\mathsf{k}} [f^{\circ}\cdot\rangle P \text{ iff } \exists x \in H(\mathcal{F}xy \text{ and } x \in_{\mathsf{h}} P).$$

where P is now a set of points in $BA(\mathsf{h})$.

A *distributed frame* has a Stone space, $\text{Stone}(\mathsf{h}) = (H, \mathcal{H}, BA(\mathsf{h}))$, at each node where H is collection of points, $\mathcal{H}$ is a endo-relation, and $BA(\mathsf{h})$ is a Boolean set algebra of points. We sometimes display the type of relations through an abuse of notation as $\mathcal{F} : \mathsf{h} \rightharpoonup \mathsf{k}$ for $\mathcal{F} \subseteq H \times K$ where the direction is by fiat from H to K. This abuse is justified since there will be a relation in the distributed frame for every arc in the graph of the logic. In general, the lower case letters in modal connectives and in arcs of the graph are depicted using their script upper case versions in the frames.

Definition 3.2.1 (Kupke, Kurz, and Venema [3]) A *general frame* is a structure $(H, \mathcal{H}, BA(\mathsf{h}))$ such that $(H, \mathcal{H})$ is a Kripke frame and $BA(\mathsf{h})$ contains $BA(H)$ which is collection of so-called *admissible* subsets of H that is closed under the Boolean operations and under the operation $[h^{\Box}\rangle : BA(\mathsf{h}) \to BA(\mathsf{h})$ given by:

$$[h^{\Box}\rangle P \stackrel{def}{=} \{x \in X \mid \mathcal{H}xy \text{ implies } y \in_{\mathsf{h}} P\}$$

with $[h^{\circ}\rangle P = \neg[h^{\Box}\rangle \neg P$ where $\neg$ is set complement. A general frame $\mathcal{H}$ is called *differentiated* if for all distinct $x, y \in H$ there is a 'witness' $P \in BA(\mathsf{h})$ such that $y \in_{\mathsf{h}} P$ while $x \notin_{\mathsf{h}} P$; *tight* if whenever y is not an $\mathcal{H}$-successor of x, then there is a 'witness' $P \in_{\mathsf{h}} BA(\mathsf{h})$ such that $y \in a$ while $x \notin_{\mathsf{h}} [h^{\circ}\rangle P$; and *compact* if $\bigcap A \neq \emptyset$ for every subset A of $BA(\mathsf{h})$ which has the finite intersection property. A general frame is *descriptive* if it is differentiated, tight, and compact. □

As it is noted in [3], tightness can be re-expressed as a relation being point closed, i.e., $\mathcal{H}x$ (equal to $\mathcal{H}\{x\}$), the forward image of x under $\mathcal{H}$, is a closed set in the topology generated by the algebra $BA(\mathsf{h})$ of clopen sets.

A good mental picture to remember the definition of a frame is the following diagram where P is either proposition of a language or a UCLA proposition. This latter is merely a set of points in a lattice of sets. In the former case, the $\in_{\mathsf{h}}$ relation becomes $\models^{\mathsf{h}}$ and in fact, in interpretations, $\in_{\mathsf{h}}$ is the meaning of $\models^{\mathsf{h}}$.

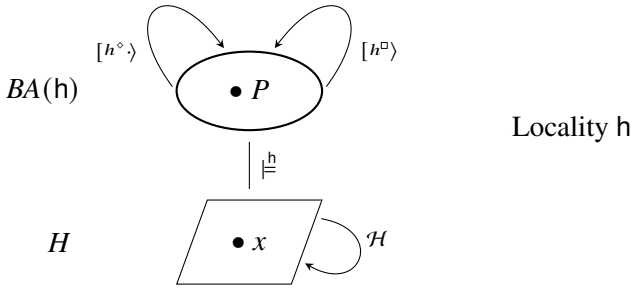

Figure 3.1: Intuitive Diagram of Locality at h

Here x is some point in the domain H and P is either a proposition or some UCLA proposition in $BA(\mathsf{h})$, but it is not necessarily the case that $x \models^{\mathsf{h}} P$. Two example modal operators $[h^{\Box}\rangle$ and $[h^{\diamond}\cdot\rangle$ are shown. The relation $\mathcal{H} : \mathsf{h} \rightharpoonup \mathsf{h}$ is some set $\mathcal{H} \subseteq H \times H$. Note that h refers to the node at which the frame indicated by the diagram lives. The h in $[h^{\Box}\rangle$ and $[h^{\diamond}\rangle$ refers to the relation $\mathcal{H}$ in any interpretation.

We use one category of DG frames for each Distributed Logic. The Section 8.1 of the Appendix contains a more complete and formal presentation of DL from [4].

3.3 Distribution

The distributed operators take propositions in one locality (component) and translate them to propositions in another. A distributed frame for modeling two components has two localities. The only restriction on the graph of localities is that it be finite. The reason for this is that the graph is part of the formal logic. In the future, we may wish to revisit this decision, but the applications for such an infinite graph are likely to be very abstract and mathematically based.

The next diagram depicts this situation (without showing the local modalities or local relations) and two (generic) distributed modalities:

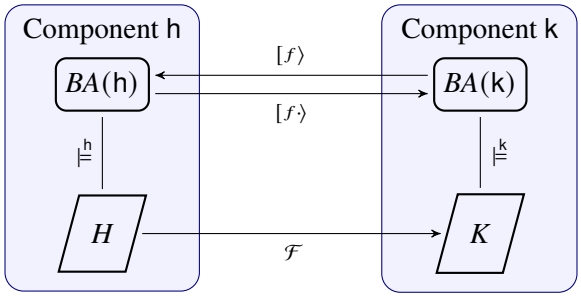

Figure 3.2: Generic Distribution with Two Components as Localities

where the arrows $[f\rangle$, $[f\cdot\rangle$ can be any of the modal forward and backward distributed modalities interpreted by the distributed Kripke relation $\mathcal{F}$, i.e., the lower case f in the modalities is linked with the script $\mathcal{F}$ relation. Example pairs of modalities are $[f^{\Box}\rangle$ and $[f^{\Box}\cdot\rangle$ and are necessity operators, and $[f^{\diamond}\rangle$ and $[f^{\diamond}\cdot\rangle$ are possibility operators.

The diagram is only showing the simplest of distributed frames. In any application, even one with only two localities, there can be any number of distributed relations and any number of distributed modalities.

Definition 3.3.1 A *distributed frame category* has a descriptive, general frame, called a *local frame*, for each node in the graph, and a point closed relation for each arrow. That is, for $f : \mathsf{h} \rightharpoonup \mathsf{k}$ in the graph, f's interpretation $\mathcal{F} : \mathcal{H} \rightharpoonup \mathcal{K}$ must be point closed. □

The corresponding Kripke frame conditions for the logical axioms are

Frame Conditions S:

FS1. A category of *local modal frames* and simulations

FS2. An endo-arrow $\mathcal{I}_{\mathsf{h}}$ for the arc $i_{\mathsf{h}} \in \mathfrak{G}$

Frame Conditions A: For each node in $\mathfrak{G}$,

FA1. A set of classical worlds

FA2. Modal frame conditions for a logic at this node

FA3. $\mathcal{I}_{\mathsf{h}}$ is an identity relation

Frame Conditions B:

FB1. Frame category contains all compositions

FB2. Frame category contains all converse compositions

with the convention that upper case script relation letters interpret modalities, the latter using lower case Roman letters. Each distributed frame category interpreting a DL will have the conditions matching the axioms. The frame conditions **S**, **A**, and **B** are always assumed, additional axioms require additional frame conditions.

The distributed relation $\mathcal{F} : \mathsf{h} \rightharpoonup \mathsf{k}$ used in the evaluation of the operators above uses two localities, h and k. Here, h and k are variables in that any actual FPGA application will fill in those localities by two components; the entire distributed frame will use as many localities as there are components and possibly more if individual processes within a component must be singled out for analysis. Also, any one locality may have many local relations, and any two localities may be connected by many distributed relations. The modalities involved are restricted by the number of relations we have at our disposal to interpret the modalities. Technically, $\mathcal{F} \subseteq H \times K$, we leave this implicit in the notation $\mathcal{F} : \mathsf{h} \rightharpoonup \mathsf{k}$.

The distributed relation $\mathcal{F}$ (see Figure 3.2) is denoted with an arrow but this is mere convention; $\mathcal{F}$ is a two-place relation that, by fiat, is viewed as a morphism from elements of its first position to elements of its second position. The distributed modalities, on the other hand, really are functions although they have special properties required for us to treat them as modalities.

We will often use the locution "P in h" to mean P is a proposition at locality h and hence $P \in BA(\mathsf{h})$. A similar statement holds for "Q in k".

4. USING DISTRIBUTED LOGIC IN SPECIFICATIONS

We will relate points at locality h and points at k by the relation $\mathcal{F} : \mathsf{h} \rightharpoonup \mathsf{k}$. The relation $\breve{\mathcal{F}}$ is the converse of the relation $\mathcal{F}$:

$$\breve{\mathcal{F}}yx \text{ iff } \mathcal{F}xy,$$

i.e., $\breve{\mathcal{F}}$ is the reverse of the relation $\mathcal{F}$.

Suppose we wish $\mathcal{F}$ to be a function. For the relation $\mathcal{F}$ to be a function, it must be constrained to actually be a function. The specification, for all propositions Q at k,

$$[f^{\square}\rangle\, Q \overset{\mathsf{k}}{\supset} [f^{\circ}\rangle\, Q \qquad [f^{\circ}\rangle\, Q \overset{\mathsf{k}}{\supset} [f^{\square}\rangle\, Q$$

forces the distributed Kripke relation $\mathcal{F}$ to be *entire on its domain* and *functional* respectively. Think of $[f^{\square}\rangle$ and $[f^{\circ}\rangle$ as connectives that pull back propositions Q in k to propositions in h, i.e., against the relation $\mathcal{F}$ used in their interpretations. In a similar fashion, $[f^{\square}\cdot\rangle$ and $[f^{\circ}\cdot\rangle$ pull propositions P in h from to k, i.e., against the relation $\breve{\mathcal{F}}$, or what is the same, pushes propositions P in h forward along $\mathcal{F}$ from h to k.

For $\mathcal{F}$ to be entire in its domain means that $\mathcal{F}$ is defined everywhere on its domain, i.e., all of H. For $\mathcal{F}$ to be functional means that it satisfies the characteristic property of a function in that it is single valued, i.e., if x is related to some y under $\mathcal{F}$, then $\mathcal{F}xz$ implies $y = z$ there is one and only one y such that $\mathcal{F}xy$; typically in mathematics we say that for every x there is a unique y such that $\mathcal{F}xy$ whereas here we have the caveat that x may not be related to any y under $\mathcal{F}$. The way to see that this specification forces the properties of $\mathcal{F}$ we need is via the following proofs, using the natural deduction system of [5]. The proofs are somewhat pedantic and use some shorthand so that the object language (DL) does not get confused with the model theoretic meta-language, e.g., $\supset$-Intro stands for implies-Intro

1	$a \models^{\mathsf{h}} [f^{\square}\rangle Q$	assume
2	$\forall y(\mathcal{F}ay$ implies $y \models^{\mathsf{k}} Q)$	modeling condition for $[f^{\square}\rangle$, line 1
3	$\exists z(\mathcal{F}az)$	$\mathcal{F}$ is entire over its domain
4	$\boxed{c}$ $\mathcal{F}ac$	assume
5	$\mathcal{F}ac$ implies $c \models^{\mathsf{k}} Q$	$\forall$-Elim, line 2
6	$c \models^{\mathsf{k}} Q$	$\supset$-Elim, lines 4, 5
7	$\mathcal{F}ac$ and $c \models^{\mathsf{k}} Q$	$\wedge$-Intro, lines 4, 6
8	$\exists z(\mathcal{F}az$ and $z \models^{\mathsf{k}} Q)$	$\exists$-Intro, line 7
9	$a \models^{\mathsf{h}} [f^{\circ}\rangle Q$	modeling condition for $[f^{\circ}\rangle$, line 8
10	$a \models^{\mathsf{h}} [f^{\circ}\rangle Q$	$\exists$-Elim, lines 3, 4
11	$a \models^{\mathsf{h}} [f^{\square}\rangle Q$ implies $a \models^{\mathsf{h}} [f^{\circ}\rangle Q$	$\supset$-Intro, line 1
12	$[f^{\square}\rangle P \supset^{\mathsf{h}} [f^{\circ}\rangle Q$	modeling condition for $\supset^{\mathsf{h}}$, line 11

and

1	$a \models^{\mathsf{h}} [f^{\circ}\rangle Q$	assume
2	$\exists y(\mathcal{F}ay$ and $y \models^{\mathsf{k}} Q)$	modeling condition for $[f^{\circ}\rangle$, line 1
3	$\boxed{b}$ $\mathcal{F}ab$ and $b \models^{\mathsf{k}} Q$	assume
4	$\boxed{c}$ $\mathcal{F}ac$	assume
5	$\mathcal{F}ab$	$\wedge$-Elim, line 3
6	$\mathcal{F}ab$ and $\mathcal{F}ac$	$\wedge$-Intro, lines 4, 5
7	$b = c$	$\mathcal{F}$ is functional, line 6
8	$b \models^{\mathsf{k}} Q$	$\wedge$-Elim, line 3
9	$c \models^{\mathsf{k}} Q$	=-Elim, lines 7, 8
10	$\forall y(\mathcal{F}ay$ implies $y \models^{\mathsf{k}} Q)$	$\forall$-Intro, line 4
11	$a \models^{\mathsf{h}} [f^{\square}\rangle Q$	modeling condition for $[f^{\square}\rangle$, line 10
12	$a \models^{\mathsf{h}} [f^{\square}\rangle Q$	$\exists$-Elim, lines 2, 3
13	$a \models^{\mathsf{h}} [f^{\circ}\rangle Q$ implies $a \models^{\mathsf{h}} [f^{\square}\rangle Q$	$\supset$-Intro, line 1
14	$[f^{\circ}\rangle P \supset^{\mathsf{h}} [f^{\square}\rangle Q$	modeling condition for $\supset^{\mathsf{h}}$, line 13

The specification, for all propositions P in h,

$$[f^{\square}\cdot\rangle P \supset^{\mathsf{k}} [f^{\circ}\cdot\rangle P \qquad [f^{\circ}\cdot\rangle P \supset^{\mathsf{k}} [f^{\square}\cdot\rangle P$$

forces the distributed Kripke relation converse $\breve{\mathcal{F}}$ to be *entire* on its domain and *functional* respectively. The proofs are similar now using $\breve{\mathcal{F}}$, the converse relation of $\mathcal{F}$. When the properties for $\mathcal{F}$ and $\breve{\mathcal{F}}$ hold, $\mathcal{F}$ is called a *bijection*. This means that every point at h can be paired with a unique point at k and visa versa. Put another way, the two point sets H and K have the same cardinality and the relation $\mathcal{F}$ shows how to pair up the points.

The sense in which the formulas "force" the relation to have the properties we want is that the only way the proofs can succeed is with relation $\mathcal{F}$ having the correct properties. If the formula holds as valid, then the indicated relation must have the correct properties.

4.1 Modal Composition

Suppose we have the following situation

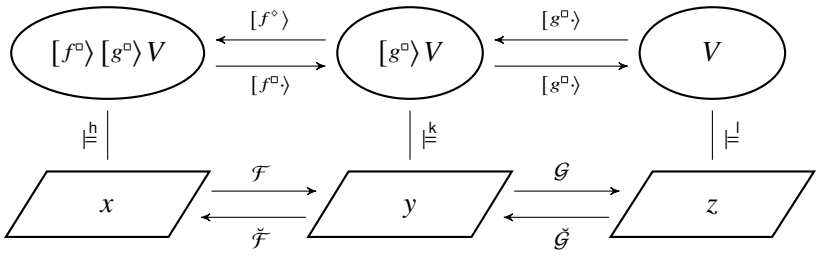

Figure 4.1: Composition of Formulas and relations

One can see from the diagram that the modal operators are applied in the opposite order as the relations are composed. Relational composition of $\mathcal{F}$ and $\mathcal{G}$ is denoted $\mathcal{F} \cdot \mathcal{G}$. As morphisms, this flips to the categorical composition $\mathcal{G} \circ \mathcal{F}$. Hence $[(f \circ g)^{\square}\rangle$ will be interpreted by the relation $\mathcal{G} \circ \mathcal{F}$ and the direction is suitably flipped between the logic and the distributed Kripke frame.

Lemma 4.1.1 *The following axioms are valid: for each $f : \mathsf{h} \rightharpoonup k$ and $g : \mathsf{k} \rightharpoonup \mathsf{l}$ in $\mathfrak{G}$,*

$$[f^{\square}]\,[g^{\square}\rangle\, U \overset{\mathsf{h}}{\equiv} [(g \circ f)^{\square}\rangle\, U \qquad [g^{\square}\cdot\rangle\,[f^{\square}\cdot\rangle\, P \overset{\mathsf{l}}{\equiv} [(f \circ g)^{\square}\cdot\rangle\, P.$$

Proof: We prove the first statement as the second is similar:

1	$\boxed{a}\quad a \in \mathsf{h}\ [f^{\square}\rangle\,[g^{\square}\rangle V$	assume
2	$\forall z\,(\mathcal{F}az$ implies $z \in \mathsf{k}\ [g^{\circ}\rangle V)$	modeling condition for $[f^{\square}\rangle$, line 1
3	$\quad\boxed{c}\quad (\mathcal{G}\circ\mathcal{F})\,ac$	assume
4	$\quad\exists z\,(\mathcal{F}az$ and $\mathcal{G}zc)$	def. $\mathcal{G}\circ\mathcal{F}$-Elim, line 3
5	$\quad\boxed{b}\quad \mathcal{F}ab$ and $\mathcal{G}bc$	assume
6	$\quad\mathcal{F}ab$	$\wedge$-Elim, line 5
7	$\quad\mathcal{F}ab$ implies $b \in \mathsf{k}\ [g^{\square}\rangle V$	$\forall$-Elim, line 2
8	$\quad b \in \mathsf{k}\ [g^{\square}\rangle V$	$\supset$-Elim, lines 6, 7
9	$\quad\forall y\,(\mathcal{G}by$ implies $y \in \mathsf{l}\ V)$	modeling condition for $[g^{\square}\rangle$, line 8
10	$\quad\mathcal{G}bc$ implies $c \in \mathsf{l}\ V$	$\forall$-Elim, line 9
11	$\quad\mathcal{G}bc$	$\wedge$-Elim, line 5
12	$\quad c \in \mathsf{l}\ V$	$\supset$-Elim, lines 10, 11
13	$c \in \mathsf{l}\ V$	$\exists$-Elim, lines 4, 5
14	$a \in \mathsf{h}\ [(g\circ f)^{\square}\rangle V$	$\supset$-Intro, line 1
15	$[f^{\square}\rangle\,[g^{\square}\rangle V \overset{\mathsf{h}}{\supset} [(g\circ f)^{\square}\rangle V$	modeling condition for $\overset{\mathsf{h}}{\supset}$, line 1

and

1	$\boxed{a}\quad a \in \mathsf{h}\ [(g\circ f)^{\square}\rangle V$	assume
2	$\forall z\,\big((\mathcal{G}\circ\mathcal{F})az$ implies $z \in \mathsf{l}\ V\big)$	modeling condition for $[(g\circ f)^{\square}\rangle$, line 1
3	$\quad\boxed{b}\quad \mathcal{F}ab$	assume
4	$\quad\boxed{c}\quad \mathcal{G}bc$	assume
5	$\quad\mathcal{F}ab \wedge \mathcal{G}bc$	$\wedge$-Intro, line 4
6	$\quad(\mathcal{G}\circ\mathcal{F})ac$ implies $c \in \mathsf{l}\ V$	$\forall$-Elim, line 2
7	$\quad c \in \mathsf{l}\ V$	$\supset$-Elim, lines 5, 6
8	$\quad\forall y\,(\mathcal{G}by$ implies $y \in \mathsf{l}\ V)$	$\forall$-Intro, line 4
9	$\quad b \in \mathsf{k}\ [g^{\square}\rangle V$	modeling condition for $[g^{\square}\rangle$, line 8
10	$\forall v\,(\mathcal{F}ay$ implies $v \in \mathsf{k}\ [g^{\square}\rangle V)$	$\forall$-Intro, line 3
11	$a \in \mathsf{h}\ [f^{\square}\rangle\,[g^{\square}\rangle V$	modeling condition for $[f^{\square}\rangle$, line 10
12	$[(g\circ f)^{\square}\rangle V \overset{\mathsf{h}}{\supset} [f^{\square}\rangle\,[g^{\square}\rangle V$	$\supset$-Intro, line 1

■

Corollary 4.1.2 *The following formulas are valid for* $f : \mathsf{h} \rightharpoonup \mathsf{k}$, $g : \mathsf{k} \rightharpoonup \mathsf{l}$, *and* $p : \mathsf{l} \rightharpoonup \mathsf{t}$,

$$[(p\circ(g\circ f))^{\square}\rangle V \overset{\mathsf{h}}{\equiv} [((p\circ g)\circ f)^{\square}\rangle V \qquad [(f\circ(g\circ p))^{\square}\cdot\rangle P \overset{\mathsf{t}}{\equiv} [((f\circ g)\circ p)^{\square}\cdot\rangle P.$$

Proof:

$$\begin{aligned}
[(p\circ(g\circ f))^{\square}\rangle V &\overset{\mathsf{h}}{\equiv} [(g\circ f)^{\square}\rangle\,[p^{\square}\rangle V \\
&\overset{\mathsf{h}}{\equiv} [f^{\square}\rangle\,[g^{\square}\rangle\,[p^{\square}\rangle V \\
&\overset{\mathsf{h}}{\equiv} [f^{\square}\rangle\,[(p\circ g)^{\square}\rangle V \\
&\overset{\mathsf{h}}{\equiv} [((p\circ g)\circ f)^{\square}\rangle V.
\end{aligned}
\qquad
\begin{aligned}
[(f\circ(g\circ p))^{\square}\cdot\rangle P &\overset{\mathsf{h}}{\equiv} [(g\circ p)^{\square}\cdot\rangle\,[f^{\square}\cdot\rangle P \\
&\overset{\mathsf{h}}{\equiv} [p^{\square}\cdot\rangle\,[g^{\square}\cdot\rangle\,[f^{\square}\cdot\rangle P \\
&\overset{\mathsf{h}}{\equiv} [p^{\square}\cdot\rangle\,[(f\circ g)^{\square}\cdot\rangle P \\
&\overset{\mathsf{h}}{\equiv} [((f\circ g)\circ p)^{\square}\cdot\rangle P.
\end{aligned}$$

■

4.2 Simulations in Distributed Logic

The simulation condition is (in a hack of the graphical notation from Freyd and Scedrov [6]). The symbol $\overset{\mathcal{H}}{\longmapsto}$ in $x \overset{\mathcal{H}}{\longmapsto} x'$ indicates a pair $\langle x, x'\rangle \in \mathcal{H}$ and is not a functional relationship. Hence the arrows in the diagram below are not arrows in a category but rather indicate pairs in binary relation.

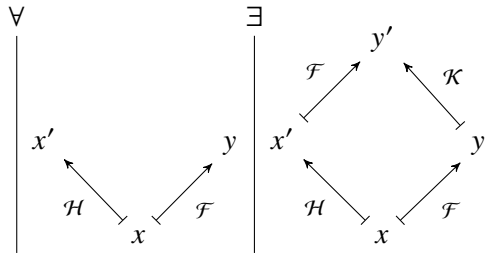

Figure 4.2: Diagrammatic Form of Simulation Condition

Formulas of the following form are from Simulation Logic [4]:

$$[f^{\circ}\rangle\,[k^{\square}\rangle\,Q \overset{\mathsf{h}}{\supset} [h^{\square}\rangle\,[f^{\circ}\rangle\,Q$$

where the distributed connective $[f^{\circ}\rangle$ has type $[f^{\circ}\rangle : \mathrm{Log}(\mathsf{k}) \to \mathrm{Log}(\mathsf{h})$ for $\mathrm{Log}(\mathsf{k})$ and $\mathrm{Log}(\mathsf{h})$, which are the logics at node k and h respectively. The connective $\overset{\mathsf{h}}{\supset}$ indicates that $\supset$ is classical implication at node h. The nodes h and k are part of the logic's graph. The local modal connectives are $[h^{\square}\rangle : \mathrm{Log}(\mathsf{h}) \to \mathrm{Log}(\mathsf{h})$ and $[k^{\square}\rangle : \mathrm{Log}(\mathsf{k}) \to \mathrm{Log}(\mathsf{k})$.

This formula is validated by the frame condition ([4], [7])

$$\mathcal{F}xy \text{ and } \mathcal{H}xx' \text{ implies } \exists y'(\mathcal{K}yy' \text{ and } \mathcal{F}x'y')$$

where $[h^{\square}\rangle$ and $[k^{\square}\rangle$ are interpreted by $\mathcal{H}$ and $\mathcal{K}$ respectively, and $[f^{\circ}\rangle$ is interpreted by $\mathcal{F}$. This formula is expresses precisely what the diagram expresses. Figure 4.3 shows the relevant localities indicating where the formulas live, what the modal operators are doing, and where the relations live:

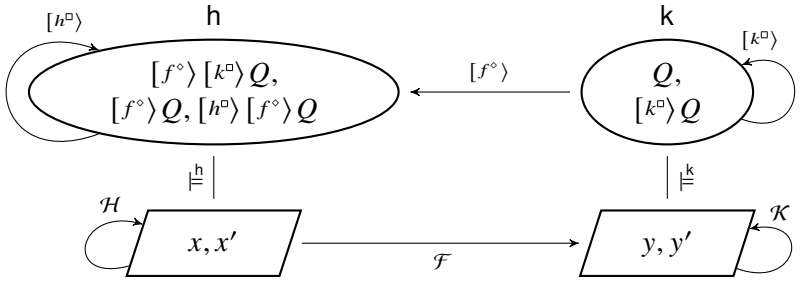

Figure 4.3: Intuitive View of Subformulas in a Simulation Axiom and its Interpretation

This usage of small italicized letters in the logic interpreted by their large italicized counterparts in the frames is carried on throughout the paper. To simplify the exposition, we generally assume a single collection of local modalities: $[h^\Box\rangle$, $[h^\circ\rangle$, $[h^\Box\cdot\rangle$, and $[h^\circ\cdot\rangle$ for two forward and two converse modalities (see [1]) at a node h. We use an interpreting relation $\mathcal{H}$ for those modalities. Local notions generally use h and k (in their various fonts), and distributed notions are similar using f and g.

A bisimulation between h and k is specified with the following formulas

$$[f^\circ\rangle\,[k^\Box\rangle\,Q \overset{\mathsf{h}}{\supset} [h^\Box\rangle\,[f^\circ\rangle\,Q \qquad [f^\circ\cdot\rangle\,[h^\Box\rangle\,P \overset{\mathsf{k}}{\supset} [k^\Box\rangle\,[f^\circ\cdot\rangle P.$$

Notice that the second formula is not a mere switching of the direction of the $\supset$ connective. The first uses the $\supset$ connective of h and the second of k. Even if only a single local logic was used, say, at h, the two formulas would not satisfy the reversal of direction of $\overset{\mathsf{h}}{\supset}$. These formula have the following validation conditions (respectively)

$$\mathcal{F}xy \text{ and } \mathcal{H}xx' \text{ implies } \exists y'(\mathcal{K}yy' \text{ and } \mathcal{F}x'y') \text{ and}$$
$$\mathcal{F}xy \text{ and } \mathcal{K}yy' \text{ implies } \exists x'(\mathcal{H}xx' \text{ and } \mathcal{F}x'y').$$

The proof that the first formula is validated by the first condition is paradigmatic of many of the simulations in the sequel and hence we repeat it here (in somewhat pedantic form) using the Fitch-style proofs of [5] and elide such validation proofs in the sequel:

1	$x \in \mathsf{h}\ [f^\circ\rangle\,[k^\Box\rangle\,Q$	assume
2	$\mathcal{F}xy$ and $y \in \mathsf{k}\ [k^\Box\rangle\,Q$	def. $[f^\circ\rangle$ for some y, line 1
3	$\mathcal{H}xx'$	assume
4	$\mathcal{F}xy$ and $\mathcal{H}xx'$	$\wedge$-Elim, $\wedge$-Intro, lines 2, 3
5	$\mathcal{K}yy'$ and $\mathcal{F}x'y'$	Simulation Condition for some y', line 4
6	$y \in \mathsf{k}\ [k^\Box\rangle\,Q$ and $\mathcal{K}yy'$	$\wedge$-Eim, $\wedge$-Intro, lines 2, 5
7	$y' \in \mathsf{k}\ Q$	def. $[k^\Box\rangle$, line 6
8	$\mathcal{F}x'y'$ and $y' \in \mathsf{k}\ Q$	$\wedge$-Eim, $\wedge$-Intro, lines 4, 7
9	$x' \in \mathsf{h}\ [f^\circ\rangle\,Q$	def. $[f^\circ\rangle$, line 8
10	$\mathcal{H}xx'$ implies $x' \in \mathsf{h}\ [f^\circ\rangle\,Q$	$\supset$-Intro, line 3
11	$x \in \mathsf{h}\ [h^\Box\rangle\,[f^\circ\rangle\,Q$	def. $[h^\Box\rangle$, line 10

The lines labeled by "def." are indicating the definition of the valuation conditions for the active formula in the step. The validation of the second formula is similar.

We collect all the forms of simulation with their sentential prescriptions:

Definition 4.2.1

Name	Frame Condition	Axiom
Forward Simulation	$\mathcal{F}xy$ and $\mathcal{H}xx'$ $\exists y'(\mathcal{K}yy'$ and $\mathcal{G}x'y')$	$[f^\circ\rangle\,[k^\square\rangle Q \overset{\mathsf{h}}{\supset} [h^\square\rangle\,[g^\circ\rangle Q$
Proverse Simulation	$\mathcal{F}x'y'$ and $\mathcal{H}xx'$ implies $\exists y(\mathcal{K}yy'$ and $\mathcal{G}xy)$	$[f^\circ\rangle\,[k^\square\cdot\rangle Q \overset{\mathsf{h}}{\supset} [h^\square\cdot\rangle\,[g^\circ\rangle Q$
Converse Simulation	$\breve{\mathcal{F}}yx$ and $\mathcal{K}yy'$ implies $\exists x'(\mathcal{H}xx'$ and $\breve{\mathcal{G}}y'x')$	$[f^\circ\cdot\rangle\,[h^\square\rangle P \overset{\mathsf{k}}{\supset} [k^\square\rangle\,[g^\circ\cdot\rangle P$
Backward Simulation	$\mathcal{F}x'y'$ and $\mathcal{K}yy'$ implies $\exists x(\mathcal{H}xx'$ and $\mathcal{G}xy)$	$[f^\circ\cdot\rangle\,[h^\square\cdot\rangle P \overset{\mathsf{k}}{\supset} [k^\square\cdot\rangle\,[g^\circ\cdot\rangle P$

Table 4.1: Frame Conditions and Axioms

□

Theorem 4.2.2 *The Frame Conditions validate their corresponding Axioms in the table.*

4.3 Summary of Axioms

We assume $\mathcal{F}, \mathcal{G} : \mathsf{h} \rightharpoonup \mathsf{k}$. If $\mathcal{G} : \mathsf{k} \rightharpoonup \mathsf{h}$, then swap $[g^\circ\rangle$ and $[g^\circ\cdot\rangle$ in the axioms. These conditions are in generalized form:

Name	Frame Condition	Axiom
Forward Simulation	$\mathcal{F}xy$ and $\mathcal{H}xx'$ implies $\exists y'(\mathcal{K}yy'$ and $\mathcal{G}x'y')$	$[f^\circ\rangle\,[k^\square\rangle Q \supset [h^\square\rangle\,[g^\circ\rangle Q$
Proverse Simulation	$\mathcal{F}x'y'$ and $\mathcal{H}xx'$ implies $\exists y(\mathcal{K}yy'$ and $\mathcal{G}xy)$	$[f^\circ\rangle\,[k^\square\cdot\rangle Q \supset [h^\square\cdot\rangle\,[g^\circ\rangle Q$
Converse Simulation	$\breve{\mathcal{F}}yx$ and $\mathcal{K}yy'$ implies $\exists x'(\mathcal{H}xx'$ and $\breve{\mathcal{G}}y'x')$	$[f^\circ\cdot\rangle\,[h^\square\rangle P \supset [k^\square\rangle\,[g^\circ\cdot\rangle P$
Backward Simulation	$\mathcal{F}x'y'$ and $\mathcal{K}yy'$ implies $\exists x(\mathcal{H}xx'$ and $\mathcal{G}xy)$	$[f^\circ\cdot\rangle\,[h^\square\cdot\rangle P \supset [k^\square\cdot\rangle\,[g^\circ\cdot\rangle P$

Table 4.2: Summary of Simulation Conditions and Axioms

Diagrammatically:

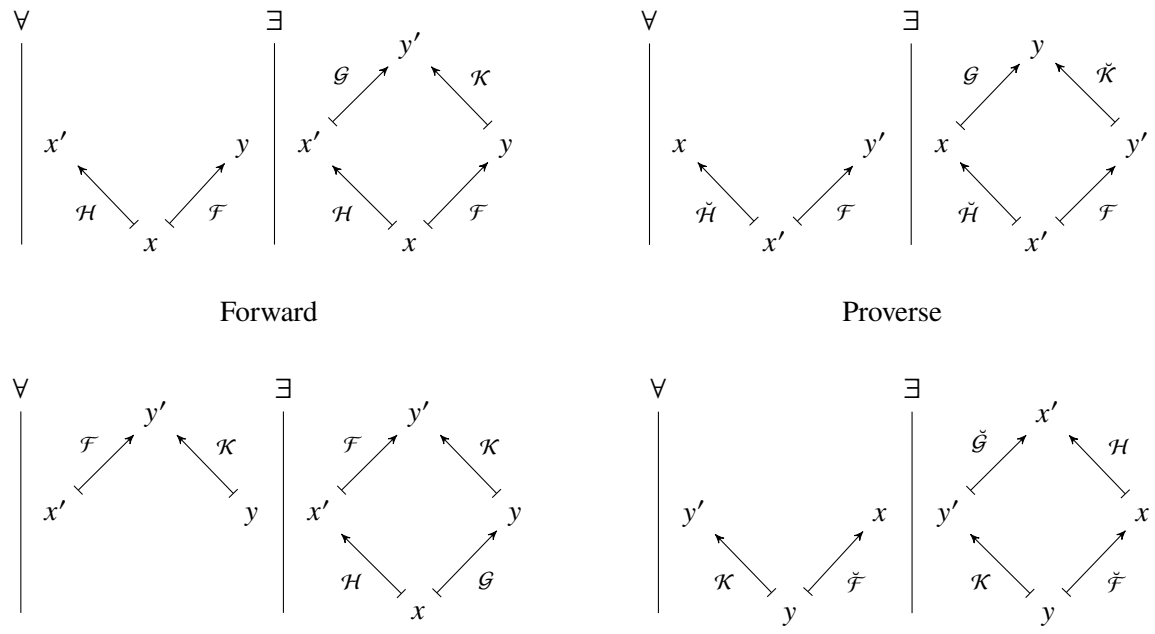

Figure 4.4: Diagrammatic Form of Simulation Conditions

5. USING FORWARD SIMULATION

We can re-express any of the simulation axioms using iterated possibilities. The composition will take place at the semantic semantic level in a category **SRel**, which is the category of Markov kernels [8] ()see also *stochastic relations* [9, 10]). Essentially this means putting probabilities on the individual pairs (arcs) of the relation. While we will not investigate Markov kernels in this paper, we expect to a future research.

5.1 Forward Simulation

$$\mathcal{F}xy \text{ and } \mathcal{H}xx' \text{ implies } \exists y'(\mathcal{K}yy' \text{ and } \mathcal{G}x'y')$$

has the diagram

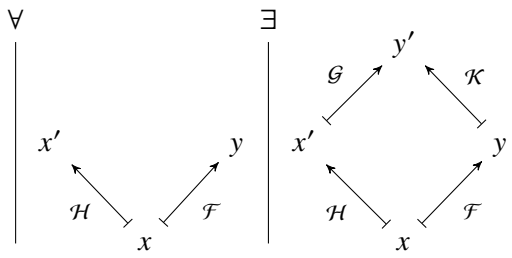

Figure 5.1: Forward Simulation

This condition validates the forward simulation axiom.

$$[f^\circ\rangle\,[k^\square\rangle\,Q \supset [h^\square\rangle\,[g^\circ\rangle\,Q$$

Lemma 5.1.1 *Let* $f : \mathsf{h} \rightarrowtail \mathsf{k}$ *in the logic's distribution graph, then the forward simulation axiom*

$$[f^\circ\rangle\,[k^\square\rangle\,Q \overset{\mathsf{h}}{\supset} [h^\square\rangle\,[g^\circ\rangle\,Q$$

is equivalent to the following two alternate forms:

$$[h^\circ\cdot\rangle[f^\circ\rangle\,S \overset{\mathsf{h}}{\supset} [g^\circ\rangle\,[k^\circ\cdot\rangle S \qquad [f^\circ\cdot\rangle[h^\circ\rangle\,P \overset{\mathsf{k}}{\supset} [k^\circ\rangle\,[g^\circ\cdot\rangle P$$

and all the forms are interderivable.

Note that in the second equivalent formula, the forward simulation axiom and this equivalent are in different localities. Recall for $f : \mathsf{h} \rightarrowtail \mathsf{k}$, that $[f^\circ\rangle$ and $[g^\circ\rangle$ pull formulas back along f and g respectively while $[f^\circ\cdot\rangle$ and $[g^\circ\cdot\rangle$ push formulas forward along f and g respectively.

The equivalent validating conditions (respectively) are

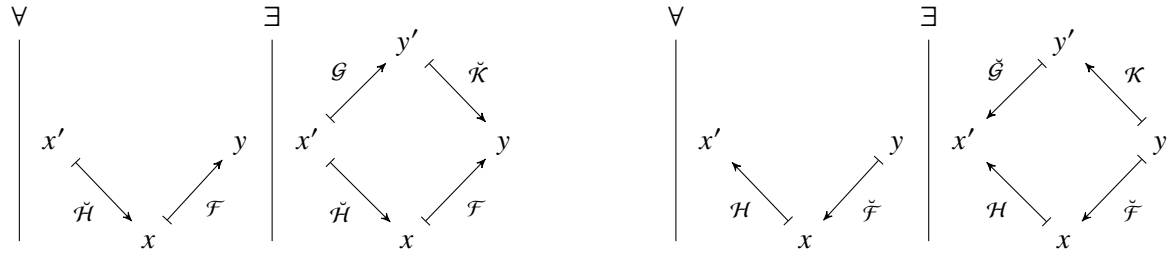

Figure 5.2: Equivalent Forward Simulation Conditions

Note the reversal of some of the arrows from the original in Figure 4.4.

Proof: The proofs requires the use of residuation rules and these are valid in the logic, for any h, k, $f : \mathsf{h} \rightarrowtail \mathsf{k}$, $P \in \mathsf{h}$ and $Q \in \mathsf{k}$:

$$\frac{P \overset{\mathsf{h}}{\supset} [f^\square\rangle\,Q}{[f^\circ\cdot\rangle P \overset{\mathsf{k}}{\supset} Q} \qquad \frac{Q \overset{\mathsf{k}}{\supset} [f^\square\cdot\rangle P}{[f^\circ\rangle\,Q \overset{\mathsf{h}}{\supset} P}$$

where the double line means the rules are bidirectional. The rules say that if the premise is provable, then the conclusion is provable (in both directions). Since the rules are valid, we can use them in their set theoretic form:

$$\frac{P \subseteq_{\mathsf{h}} [f^{\square}\rangle\, Q}{[f^{\circ}\cdot\rangle P \subseteq_{\mathsf{k}} Q} \qquad \frac{Q \subseteq_{\mathsf{k}} [f^{\square}\cdot\rangle P}{[f^{\circ}\rangle\, Q \subseteq_{\mathsf{h}} P}$$

Also, it is a theorem of DL borrowed from classical modal logic that

$$[r^{\square}\rangle\, S \equiv \neg[r^{\circ}\rangle\, \neg S \qquad [r^{\square}\cdot\rangle S \equiv \neg[r^{\circ}\cdot\rangle \neg S$$

where $r \in \{h, k, f, g\}$. The $\equiv$ sign is bi-implication. From these it is easy to show

$$\neg[r^{\square}\rangle\, S \equiv [r^{\circ}\rangle\, \neg S \qquad \neg[r^{\circ}\rangle\, S \equiv [r^{\square}\rangle\, \neg S.$$

In their set theoretic form, they become

$$\neg[r^{\square}\rangle\, S = [r^{\circ}\rangle\, \neg S \qquad \neg[r^{\circ}\rangle\, S = [r^{\square}\rangle\, \neg S.$$

All the connectives and their set theoretic operator counterparts are monotone, where in the following, $[-\rangle$ refers to any of the modal connectives:

$$\frac{P \supset S}{[-\rangle\, P \supset [-\rangle\, S}$$

One final note, the proof (below) of the equivalence of the two axioms

$$[f^{\circ}\rangle\, [k^{\square}\rangle\, Q \subseteq [h^{\square}\rangle\, [g^{\circ}\rangle\, Q \qquad [h^{\circ}\cdot\rangle [f^{\circ}\rangle\, S \subseteq [g^{\circ}\rangle\, [k^{\circ}\cdot\rangle S$$

appears to use different propositions. However, Q and S are propositional variables. In essence, they they are universally quantified over formulas. So these can be re-expressed as

$$\forall Q([f^{\circ}\rangle\, [k^{\square}\rangle\, Q \subseteq [h^{\square}\rangle\, [g^{\circ}\rangle\, Q) \qquad \forall S([h^{\circ}\cdot\rangle [f^{\circ}\rangle\, S \subseteq [g^{\circ}\rangle\, [k^{\circ}\cdot\rangle S).$$

We do not normally state axiom like this in propositional or modal logic. A similar statement holds for the second equivalence. These underwrite the formula substitutions in the proofs.

The proofs are in set theoretic form. The proof of the first equivalence is

1	$[f^{\circ}\rangle\, [k^{\square}\rangle\, Q \subseteq [h^{\square}\rangle\, [g^{\circ}\rangle\, Q$	assume
2	$[h^{\circ}\cdot\rangle [f^{\circ}\rangle\, [k^{\square}\rangle\, Q \subseteq [g^{\circ}\rangle\, Q$	residuation, line 1
3	$[h^{\circ}\cdot\rangle [f^{\circ}\rangle\, [k^{\square}\rangle\, [k^{\circ}\cdot\rangle S \subseteq [g^{\circ}\rangle\, [k^{\circ}\cdot\rangle S$	subst. $[k^{\circ}\cdot\rangle S$ for Q, line 2
4	$[k^{\circ}\cdot\rangle S \subseteq [k^{\circ}\cdot\rangle S$	identity
5	$S \subseteq [k^{\square}\rangle\, [k^{\circ}\cdot\rangle S$	residuation, line 4
6	$[h^{\circ}\cdot\rangle [f^{\circ}\rangle\, S \subseteq [h^{\circ}\cdot\rangle [f^{\circ}\rangle\, [k^{\square}\rangle\, [k^{\circ}\cdot\rangle S$	monotonicity of $[h^{\circ}\cdot\rangle$, $[f^{\circ}\rangle$, line 5
7	$[h^{\circ}\cdot\rangle [f^{\circ}\rangle\, S \subseteq [g^{\circ}\rangle\, [k^{\circ}\cdot\rangle S$	transitivity of $\subseteq$, lines 3, 6

and

1. $[h^\circ\rangle[f^\circ\rangle S \subseteq [g^\circ\rangle[k^\circ\rangle S$ . . . . . assume
2. $[f^\circ\rangle S \subseteq [h^\Box\rangle[g^\circ\rangle[k^\circ\rangle S$ . . . . . residuation, line 1
3. $[f^\circ\rangle[k^\Box\rangle Q \subseteq [h^\Box\rangle[g^\circ\rangle[k^\circ\rangle[k^\Box\rangle Q$ . . . . . subst. $[k^\Box\rangle Q$ for S, line 2
4. $[k^\Box\rangle Q \subseteq [k^\Box\rangle Q$ . . . . . identity
5. $[k^\circ\cdot\rangle[k^\Box\rangle Q \subseteq Q$ . . . . . residuation, line 4
6. $[h^\Box\rangle[g^\circ\rangle[k^\circ\cdot\rangle[k^\Box\rangle Q \subseteq [h^\Box\rangle[g^\circ\rangle Q$ . . . . . monotonicity of $[h^\Box\rangle$, $[g^\circ\rangle$, line 5
7. $[f^\circ\rangle[k^\Box\rangle Q \subseteq [h^\Box\rangle[g^\circ\rangle Q$ . . . . . transitivity of $\subseteq$, lines 3, 6

The proof of the second equivalence is

1. $[f^\circ\rangle[k^\Box\rangle Q \subseteq [h^\Box\rangle[g^\circ\rangle Q$ . . . . . assume
2. $\neg[h^\Box\rangle[g^\circ\rangle Q \subseteq \neg[f^\circ\rangle[k^\Box\rangle Q$ . . . . . contraposition
3. $[h^\circ\rangle[g^\Box\rangle\neg Q \subseteq [f^\Box\rangle[k^\circ\rangle\neg Q$ . . . . . classical modalities, line 2
4. $[f^\circ\cdot\rangle[h^\circ\rangle[g^\Box\rangle\neg Q \subseteq [k^\circ\rangle\neg Q$ . . . . . residuation, line 3
5. $[f^\circ\cdot\rangle[h^\circ\rangle[g^\Box\rangle\neg\neg[g^\circ\cdot\rangle P \subseteq [k^\circ\rangle\neg\neg[g^\circ\cdot\rangle P$ . . . . . subst. $\neg[g^\circ\cdot\rangle P$ for Q, line 4
6. $[f^\circ\cdot\rangle[h^\circ\rangle[g^\Box\rangle[g^\circ\cdot\rangle P \subseteq [k^\circ\rangle[g^\circ\cdot\rangle P$ . . . . . classical negation, line 5
7. $[g^\circ\cdot\rangle P \subseteq [g^\circ\cdot\rangle P$ . . . . . identity
8. $P \subseteq [g^\Box\rangle[g^\circ\cdot\rangle P$ . . . . . residuation, line 7
9. $[f^\circ\cdot\rangle[h^\circ\rangle P \subseteq [f^\circ\cdot\rangle[h^\circ\rangle[g^\Box\rangle[g^\circ\cdot\rangle P$ . . . . . monotonicity of $[f^\circ\cdot\rangle$, $[h^\circ\rangle$, line 8
10. $[f^\circ\cdot\rangle[h^\circ\rangle P \subseteq [k^\circ\rangle[g^\circ\cdot\rangle P$ . . . . . transitivity of $\subseteq$, lines 6, 9

and

1. $[f^\circ\cdot\rangle[h^\circ\rangle P \subseteq [k^\circ\rangle[g^\circ\cdot\rangle P$ . . . . . assume
2. $\neg[k^\circ\rangle[g^\circ\cdot\rangle P \subseteq \neg[f^\circ\cdot\rangle[h^\circ\rangle P$ . . . . . contraposition, line 1
3. $[k^\Box\rangle[g^\Box\cdot\rangle\neg P \subseteq [f^\Box\cdot\rangle[h^\Box\rangle\neg P$ . . . . . classical modalities, line 2
4. $[f^\circ\rangle[k^\Box\rangle[g^\Box\cdot\rangle\neg P \subseteq [h^\Box\rangle\neg P$ . . . . . residuation, line 3
5. $[f^\circ\rangle[k^\Box\rangle[g^\Box\cdot\rangle\neg\neg[g^\circ\cdot\rangle Q \subseteq [h^\Box\rangle\neg\neg[g^\circ\cdot\rangle Q$ . . . . . subst. $\neg[g^\circ\cdot\rangle Q$ for P, line 4
6. $[f^\circ\rangle[k^\Box\rangle[g^\Box\cdot\rangle[g^\circ\cdot\rangle Q \subseteq [h^\Box\rangle[g^\circ\cdot\rangle Q$ . . . . . classical negation, line 5
7. $[g^\circ\cdot\rangle Q \subseteq [g^\circ\cdot\rangle Q$ . . . . . identity
8. $Q \subseteq [g^\Box\rangle[g^\circ\cdot\rangle Q$ . . . . . residuation, line 7
9. $[f^\circ\rangle[k^\Box\rangle Q \subseteq [f^\circ\rangle[k^\Box\rangle[g^\Box\rangle[g^\circ\cdot\rangle Q$ . . . . . monotonicity of $[f^\circ\rangle$, $[k^\Box\rangle$, line 8
10. $[f^\circ\rangle[k^\Box\rangle Q \subseteq [h^\Box\rangle[g^\circ\rangle Q$ . . . . . transitivity of $\subseteq$, lines 6, 9

■

The axioms can be put in 2-diagrammatic form where the single arrows are relations, composition of arrows is relational composition, and the double arrow in the middle is interpreted as $\subseteq$. Binary relations and Stone spaces constitute a category. Categorical composition uses the symbol $\circ$ and is defined in terms of

relational composition; the latter uses the symbol $\cdot$. Composition is defined for $\mathcal{R} : X \rightharpoonup Y$ and $\mathcal{S} : Y \rightharpoonup Z$ with

$$\begin{aligned}(\mathcal{S} \circ \mathcal{R})xz \text{ iff } &(\mathcal{R} \cdot \mathcal{S})xz \\ \text{iff } &x(\mathcal{R} \cdot \mathcal{S})z \\ \text{iff } &\exists y \in Y(\mathcal{R}xy \text{ and } \mathcal{S}yz).\end{aligned}$$

where in this instance we use the alternate depiction of a relational pair in infix for the second iff. As an example, the diagrammatic form of proverse simulation is then

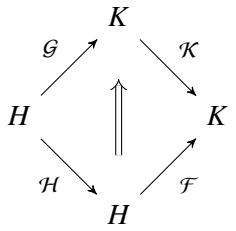

Figure 5.3: Proverse Simulation

which says that

$$\mathcal{F} \circ \mathcal{H} = \mathcal{K} \circ \mathcal{G}.$$

In relational composition, this matches with the proverse axiom, i.e.,

$$\mathcal{H} \cdot \mathcal{F} = \mathcal{G} \cdot \mathcal{K} \qquad [h^\circ\rangle\,[f^\circ\rangle\,Q \subseteq [g^\circ\rangle\,[k^\circ\rangle\,Q.$$

6. TARGETING FPGAS

This is a preliminary approach to adding just enough extra machinery to Distributed Logic to handle FPGA applications. The basic model is here but it not yet applied to any specific FPGA application. We must still learn how to use this basic model to handle security statements.

6.1 Points and States

There is a delicate matter with respect to states and FPGAs. Engineers typically use state machines where the state variable is not the usual philosophy nor the science notion of state. The latter, for which we use the term "point", is a "cut" through the world or system that settles all the relevant values of variables and constants. The former is a weaker notion because what is needed are states used in case statements for finite machine based control. Technically, in model theoretic terms, a point is a maximal filter that settles all relevant values of variables. This means that a change in any one value of a state indicates a different state. Here, the variables are thought of a vector where the addresses are variables and the values are the contents of that variable in the vector.

An engineer's notion of state is best represented as a mere filter that is not maximal. Let s be a state variable in an FPGA application and v be some value it can be assigned. To interpret this in a frame we need to expand the notion of local Kripke frame a bit.

Definition 6.1.1 A flat partial order $\leq\colon S \rightharpoonup H$ between a set S and a set H is a partial order where $S \cap H = \emptyset$. □

A depiction of a flat partial order is for $s \in S$ and $x, y, \ldots \in H$ is

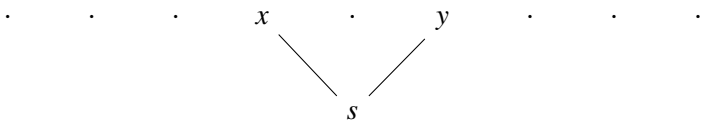

Figure 6.1: Flat Partial Order

For an FPGA application with only a single state machine, one can view $x \in H$ as settling the values of all application variables. An element $s \in S$ settles the value of only the state variable. Let the state variable in the application be state. Hence every $x \in H$ has a value for state. We will depict the value v of state at s as $s[\mathsf{state}] = v$ Let $x[p] = v$ indicate the value of the application variable p be v at point x.

We define a flat partial order $\leq\colon S \rightharpoonup H$ with

$$s \leq x \text{ iff } s[\mathsf{state}] = x[\mathsf{state}].$$

Treating states this way does NOT indicate we can represent a state s as the collection of points it is underneath in the flat domain, i.e.,

$$\bar{s} = \{x \mid s \leq x\}.$$

The reason is that there may be some x for which $s \leq x$ yet x sets the values of some application variables but those values can never arise in that particular state simply because of the way the finite state machine is structured.

In an observation,

Lemma 6.1.2

$$\forall s, q \in S(s \neq q \text{ implies } \bar{s} \cap \bar{q} = \emptyset) \qquad \text{and} \qquad \bigcup\{\bar{s} \mid s \in S\} = H.$$

Proof: The first condition says that the state variable in the application cannot be assigned two different values at the same time. Were this to occur, then there would be a point x such that $x \in \bar{s} \cap \bar{q}$. Let $x[\mathsf{state}] = v = s[\mathsf{state}]$. Since $s[\mathsf{state}] \neq q[\mathsf{state}]$, then $x \notin \bar{q}$, a contradiction.

The second says there every point x must determine the value of state. Hence there is some s such that $s[\mathsf{state}] = x[\mathsf{state}]$, and $s \leq x$ and $x \in \bar{s}$.

■

Since there may be points that we do not wish to consider as being compatible with a particular s. This can occur if $x[p] = v$ for some application variable p and $s[\mathsf{state}] = x[\mathsf{state}]$, yet p can never have the value v when the machine is in state s simply because of the structure of the machine. In a sense, $\leq$ is too large. A weaker partial order $\preceq$ taking this into account would be a subrelation of $\leq$ satisfying

$$s \preceq x \text{ implies } s \leq x.$$

We will modify the definition of a Kripke frame to the following:

Definition 6.1.3 An *FPGA frame* for a single finite state machine with state variable state is a structure $(H, S, \mathcal{H}, BA(\mathsf{h}))$ at a node h such that $(H, \mathcal{H}, BA(\mathsf{h}))$ is a general frame, and there is a flat partial order $\preceq$: $S \rightharpoonup H$ where $S \cap H = \emptyset$ and for $s \in S$ and $x, y \in H$,

$$s \preceq x \text{ implies } s[\mathsf{state}] = x[\mathsf{state}].$$

□

We can get a representation using the weaker partial order:

$$\tilde{s} = \{x \mid s \preceq x\}.$$

This has the effect of assigning a weight to a state as its set of points.

Think of points as being lines of a truth table. Each point specifies the values of all the columns where the columns are application variables. Some lines are inadmissible simply because the state machine is structured in such a way that certain values for variables can never occur under the design. This, of course, does not mean they never occur as there may be hardware errors that crop up. The admissible lines yield a theory of operation.

In [11], they observe that missing lines of a truth table correspond to regularities in the theory of a system. The notion of a "regularity" comes from system theory and indicates that there are relationships among sets of propositions. Their missing lines are our inadmissible lines. We would like to consider the inadmissible lines as being error conditions that a spider might check. They will crop up when we begin assigning probabilities to propositions.

A simple error proposition is the conjoined columns of a truth table in the guise of

$$p \stackrel{def}{=} (p_0 = a_0) \wedge (p_1 = a_1) \wedge \cdots \wedge (p_{n-1} = a_{n-1}).$$

The conjunction is finite because FPGA applications are necessarily defined via a finite description with finite quantifiers. If this proposition is measured to have a significant probability of occurrence, then there is something wrong with the design. It is considered wrong in the context of the application. In this sense, this proposition is not part of the theory of the application as expressed in the design.

In the lattice of propositions, p occurs somewhere. Using $\leq$ as the order on that lattice that can be defined $p \leq p'$ iff $p' = p \vee p'$. Let

$$s = \uparrow p = \{p'' \mid p \leq p''\},$$

then x is a principle filter on that lattice. In a broader context where the lattice is merely some lattice of propositions and not necessarily finite, let x be any maximal filter in the lattice and containing an error proposition p. In this case,

$$s \leq x \text{ and } s \not\leq x.$$

In the lattice of propositions, s is a principle filter, and both $s \leq x$ and $s \leq x$ imply $s \subseteq x$. Typically in algebraic logic, only the maximal filters are used. Introducing the principle filters will require some alteration of DL to treat the state propositions, e.g., state = s_flush, as special so they can be used to generate the states in the canonical model.

Any collection of points is considered a UCLA proposition. The power set of points will also include sets of points that are not determined by a state. Let a set of points C_s be determined by a state s if

$$C_s = \tilde{s}.$$

We can call these sets of points the *state propositions* and the collection of all of them C, i.e.

$$C = \{C_s \mid s \in S\}.$$

If $\mathcal{P}H$ is the powerset lattice of sets of H, then $C \subseteq \mathcal{P}H$. That $C \neq \mathcal{P}H$ means there are collections of states that fall into two categories, those that are consistent with the states and those that are inconsistent. By that we mean

Definition 6.1.4 B is *consistent with* S if

$$B \subseteq \bigcup_{s \in S} \{\tilde{s}\}.$$

Similarly, B is consistent with s if $B \subseteq \tilde{s}$. B is *inconsistent with* S if for no $s \in S$ is it the case that $B \subseteq \tilde{s}$. Expressed a bit more mathematically, this is

$$B \not\subseteq \bigcup_{s \in S} \{\tilde{s}\}.$$

There must be some $x \in B$ such that $x \notin \bigcup_{s \in S} \{\tilde{s}\}$. In other words, there must be some $x \in B$ such that for all $s \in S$, we have $s \not\leq x$, i.e., for all $s \in S$, we have $B \not\subseteq \tilde{s}$. □

So we want to consider the entire set lattice as the power set of X, which contains many sets in addition to the consistent sets .

7. CONCLUSION

This report is intended to set out basic Distributed Logic that is useful for Spiders. In particular, the last Chapter is for use with FPGA applications. The main type of security statements detailed in this report are simulation statements of various kinds. The Appendix shows all the possible additional normal, binary Galois operators (distributed operators) that may be of used in security statements. There are many non-normal distributed operators possible and their properties have not yet been explored for their use in security statements.

Currently, the simulation statements use the possibility and necessary operators. We must still do research to learn what kinds of simulation type statements are available using the Galois operators. The general distributed frames will certainly carry over for use with these operators.

8. APPENDIX

8.1 Distributed Logic

From [4], the axiom set for the base Simulation Logic is

Graph

S3. A directed graph $\mathfrak{G}$ of nodes and arrows

S4. An endo-arrow i_{h} for each node in $\mathfrak{G}$

Axiom Schemes A: For each node in $\mathsf{h} \in \mathfrak{G}$, and endo-arrow i_{h} at h,

A1. all truth functional theorems of a propositional logic at h

A2. Normal modal axioms for a local logic at h

A3. $P \overset{\mathsf{h}}{\equiv} [i_{\mathsf{h}}^{\Box}\rangle\, P$

Axiom Schemes B: For each arrow $f : \mathsf{h} \rightharpoonup \mathsf{k}$ and $g : \mathsf{k} \rightharpoonup \mathsf{l}$ in $\mathfrak{G}$,

B1. $[f^{\Box}\rangle\, [g^{\Box}\rangle\, V \overset{\mathsf{h}}{\equiv} [f \circ g^{\Box}\rangle\, V$

B2. $[g^{\Box}\cdot\rangle\, [f^{\Box}\cdot\rangle\, P \overset{\mathsf{h}}{\equiv} [f \circ g^{\Box}\cdot\rangle\, P$

The following Axioms are optional and we do not assume them. We add them here to give a flavor of what is possible.

Axiom Schemes D: To force arrows to be functions, use the following **D** axioms, just as they would in any normal modal logic systems. We do not assume these axioms hold in the sequel:

D1. $[f^{\Box}\rangle\, Q \overset{\mathsf{h}}{\supset} [f^{\diamond}\rangle\, Q$

D2. $[f^{\diamond}\rangle\, Q \overset{\mathsf{h}}{\supset} [f^{\Box}\rangle\, Q$

Axiom Schemes E: The axiom E1 is only necessary if you wish the classical proposition logic at $\mathrm{Log}(\mathsf{h})$ to be included in the logic at $\mathrm{Log}(\mathsf{k})$. It is not strictly necessary although it does pick up the clause in the definition of simulation [7] requiring this of a simulation. We do not assume these axioms hold in the sequel:

For all propositional letters p,

E1. $p \overset{\mathsf{h}}{\supset} [f^{\Box}\rangle\, p$ E2. $p \overset{\mathsf{k}}{\supset} [f^{\Box}\cdot\rangle p$

We assume each locality h has at least a modal relation $\mathcal{H}$ to interpret formulas at $\mathrm{Log}(\mathsf{h})$.

Definitions and Rules (Normal Systems)

Definition of Possibility: $[m^{\diamond}\rangle\, P \overset{def}{=} \neg[m^{\Box}\rangle \neg P, \quad m \in \{h, k, f\}$

Local Rules: For a local logic at h,

$$\frac{\overset{\mathsf{h}}{\vdash} P \qquad \overset{\mathsf{h}}{\vdash} P \supset R}{\overset{\mathsf{h}}{\vdash} R} \qquad \frac{\overset{\mathsf{h}}{\vdash} (P_1 \wedge \cdots \wedge P_n) \supset P}{\overset{\mathsf{h}}{\vdash} ([h^{\Box}\rangle\, P_1 \wedge \cdots \wedge [k^{\Box}\rangle\, P_n) \supset [k^{\Box}\rangle\, P}$$

The second rule is a bit redundant with respect to the normal modal logic axioms A2 since it will force any local logic to be normal.

Distributed Rules For each $f : \mathsf{h} \rightharpoonup \mathsf{k}$ arrow in $\mathfrak{G}$,

$$\frac{\overset{\mathsf{k}}{\vdash} (Q_1 \wedge \cdots \wedge Q_n) \supset Q}{\overset{\mathsf{h}}{\vdash} ([f^{\Box}\rangle\, Q_1 \wedge \cdots \wedge [f^{\Box}\rangle\, Q_n) \supset [f^{\Box}\rangle\, Q} \qquad \frac{\overset{\mathsf{h}}{\vdash} (P_1 \wedge \cdots \wedge P_n) \supset P}{\overset{\mathsf{k}}{\vdash} ([f^{\Box}\cdot\rangle P_1 \wedge \cdots \wedge [f^{\Box}\cdot\rangle P_n) \supset [f^{\Box}\cdot\rangle P}$$

8.2 Galois Operators

There are some additional normal, modal connectives in Distributed Logic that may become useful in FPGA applications. The list is complete simply because any prospective additions would be equivalent to those listed here for technical reasons we will not go into. These are termed *Galois Operators* because of their residuation properties. The legends on the left, e.g., $\vee\mapsto\wedge$, tell us the distribution properties of the operators of operators, e.g., $[f^{\perp}\cdot\rangle(P \vee P') \equiv [f^{\perp}\cdot\rangle P \wedge [f^{\perp}\cdot\rangle P'$. These are syntactic properties. The table give us the semantic interpretation conditions and the method for defining their canonical relations.

Table of all Two-Place Galois Operators

Menu A

$\wedge\mapsto\wedge$ $y \in_{\mathsf{k}} [f^{\flat}\cdot\rangle P$ iff $\forall x(x \in_{\mathsf{h}} P$ or $\mathcal{F}^{\flat\cdot}xy)$
$\mathcal{F}^{\flat\cdot}xy$ iff $\exists a(a \notin_{\mathsf{h}} x$ and $[f^{\flat}\cdot\rangle a \in_{\mathsf{k}} y)$

$\vee\mapsto\vee$ $y \in_{\mathsf{k}} [f^{\circ}\cdot\rangle P$ iff $\exists x(x \in_{\mathsf{h}} P$ and $\mathcal{F}^{\circ\cdot}xy)$
$\mathcal{F}^{\circ\cdot}xy$ iff $\forall a(a \notin_{\mathsf{h}} x$ or $[f^{\circ}\cdot\rangle a \in_{\mathsf{k}} y)$

$\vee\mapsto\wedge$ $y \in_{\mathsf{k}} [f^{\perp}\cdot\rangle P$ iff $\forall x(x \notin_{\mathsf{h}} P$ or $\mathcal{F}^{\perp\cdot}xy)$
$\mathcal{F}^{\perp\cdot}xy$ iff $\exists a(a \in_{\mathsf{h}} x$ and $[f^{\perp}\cdot\rangle a \in_{\mathsf{k}} y)$

$\wedge\mapsto\vee$ $y \in_{\mathsf{k}} [f^{?}\cdot\rangle P$ iff $\exists x(x \notin_{\mathsf{h}} P$ and $\mathcal{F}^{?\cdot}xy)$
$\mathcal{F}^{?\cdot}xy$ iff $\forall a(a \in_{\mathsf{h}} x$ or $[f^{?}\cdot\rangle a \in_{\mathsf{k}} y)$

Menu B

$\wedge\mapsto\wedge$ $y \in_{\mathsf{k}} [f^{\square}\cdot\rangle P$ iff $\forall x(x \in_{\mathsf{h}} P$ or $\neg\mathcal{F}^{\square\cdot}xy)$
$\mathcal{F}^{\square\cdot}xy$ iff $\forall a(a \in_{\mathsf{h}} x$ or $[f^{\square}\cdot\rangle a \notin_{\mathsf{k}} y)$

$\vee\mapsto\vee$ $y \in_{\mathsf{k}} [f^{\sharp}\cdot\rangle P$ iff $\exists x(x \in_{\mathsf{h}} P$ and $\neg\mathcal{F}^{\sharp\cdot}xy)$
$\mathcal{F}^{\sharp\cdot}xy$ iff $\exists a(a \in_{\mathsf{h}} x$ and $[f^{\sharp}\cdot\rangle a \notin_{\mathsf{k}} y)$

$\vee\mapsto\wedge$ $y \in_{\mathsf{k}} [f^{!}\cdot\rangle P$ iff $\forall x(x \notin_{\mathsf{h}} P$ or $\neg\mathcal{F}^{!\cdot}xy)$
$\mathcal{F}^{!\cdot}xy$ iff $\forall a(a \notin_{\mathsf{h}} x$ or $[f^{!}\cdot\rangle a \notin_{\mathsf{k}} y)$

$\wedge\mapsto\vee$ $y \in_{\mathsf{k}} [f^{\star}\cdot\rangle P$ iff $\exists x(x \notin_{\mathsf{h}} P$ and $\neg\mathcal{F}^{\star\cdot}xy)$
$\mathcal{F}^{\star\cdot}xy$ iff $\exists a(a \notin_{\mathsf{h}} x$ and $[f^{\star}\cdot\rangle a \notin_{\mathsf{k}} y)$

Menu C

$\wedge\mapsto\wedge$ $x \in_{\mathsf{h}} [f^{\flat}\rangle Q$ iff $\forall y(y \in_{\mathsf{k}} Q$ or $\mathcal{F}^{\flat}xy)$
$\mathcal{F}^{\flat}xy$ iff $\exists b(b \notin_{\mathsf{k}} y$ and $[f^{\flat}\rangle b \in_{\mathsf{h}} x)$

$\vee\mapsto\vee$ $x \in_{\mathsf{h}} [f^{\circ}\rangle Q$ iff $\exists y(y \in_{\mathsf{k}} Q$ and $\mathcal{F}^{\circ}xy)$
$\mathcal{F}^{\circ}xy$ iff $\forall b(b \notin_{\mathsf{k}} y$ or $[f^{\circ}\rangle b \in_{\mathsf{h}} x)$

$\vee\mapsto\wedge$ $x \in_{\mathsf{h}} [f^{\perp}\rangle Q$ iff $\forall y(y \notin_{\mathsf{k}} Q$ or $\mathcal{F}^{\perp}xy)$
$\mathcal{F}^{\perp}xy$ iff $\exists b(b \in_{\mathsf{k}} y$ and $[f^{\perp}\rangle b \in_{\mathsf{h}} x)$

$\wedge\mapsto\vee$ $x \in_{\mathsf{h}} [f^{?}\rangle Q$ iff $\exists y(y \notin_{\mathsf{k}} Q$ and $\mathcal{F}^{?}xy)$
$\mathcal{F}^{?}xy$ iff $\forall b(b \in_{\mathsf{k}} y$ or $[f^{?}\rangle b \in_{\mathsf{h}} x)$

Menu D

$\wedge\mapsto\wedge$ $x \in_{\mathsf{h}} [f^{\square}\rangle Q$ iff $\forall y(y \in_{\mathsf{k}} Q$ or $\neg\mathcal{F}^{\square}xy)$
$\mathcal{F}^{\square}xy$ iff $\forall b(b \in_{\mathsf{k}} y$ or $[f^{\square}\rangle b \notin_{\mathsf{h}} x)$

$\vee\mapsto\vee$ $x \in_{\mathsf{h}} [f^{\sharp}\rangle Q$ iff $\exists y(y \in_{\mathsf{k}} Q$ and $\neg\mathcal{F}^{\sharp}xy)$
$\mathcal{F}^{\sharp}xy$ iff $\exists b(b \in_{\mathsf{k}} y$ and $[f^{\sharp}\rangle b \notin_{\mathsf{h}} x)$

$\vee\mapsto\wedge$ $x \in_{\mathsf{h}} [f^{!}\rangle Q$ iff $\forall y(y \notin_{\mathsf{k}} Q$ or $\neg\mathcal{F}^{!}xy)$
$\mathcal{F}^{!}xy$ iff $\forall b(b \notin_{\mathsf{k}} y$ or $[f^{!}\rangle b \notin_{\mathsf{h}} x)$

$\wedge\mapsto\vee$ $x \in_{\mathsf{h}} [f^{\star}\rangle Q$ iff $\exists y(y \notin_{\mathsf{k}} Q$ and $\neg\mathcal{F}^{\star}xy)$
$\mathcal{F}^{\star}xy$ iff $\exists b(b \notin_{\mathsf{k}} y$ and $[f^{\star}\rangle b \notin_{\mathsf{h}} x)$

REFERENCES

1. J. M. Dunn and G. Hardegree, *Algebraic Methods in Philosophical Logic*, Oxford Logic Guides 41 (Oxford University Press, 2001).

2. G. Allwein and W. L. Harrison, "Distributed Modal Logic," in K. Bimbó, ed., *J. Michael Dunn on Information Based Logic: Outstanding Contributions to Logic*, pp. 331–362 (Springer-Verlag, 2016).

3. C. Kupke, A. Kurz, and Y. Venema, "Stone Coalgebras," in H. P. Gumm, ed., *Coalgebraic Methods in Computer Science, Electronic Notes in Theoretical Computer Science*, volume 82 of *1*, 2003, pp. 170–190.

4. G. Allwein, W. L. Harrison, and D. Andrews, "Simulation logic," *Logic and Logical Philosophy* **23**(3), 277–299 (2014).

5. D. Barker-Plummer, J. Barwise, and J. Etchemendy, *Language, Proof, and Logic*, second ed. (CSLI Publications, 2011).

6. P. J. Freyd and A. Scedrov, *Categories, Allegories* (North–Holland, 1990).

7. P. Blackburn, M. de Rijke, and Y. Venema, *Modal Logic* (Cambridge University Press, 2001). Cambridge Tracts in Theoretical Computer Science, No. 53.

8. P. Panangaden, "The Category of Markov Kernels," *Electronic Notes in Theoretical Computer Science* **22**, 171–187 (1999).

9. E. E. Doberkat, *Stochastic Relations: Foundations for Markov Transition Systems* (Chapman and Hall/CRC Studies in Informatics Series, 2007).

10. E. E. Doberkat, *Stochastic Coalgebraic Logic* (Springer-Verlag, 2010).

11. J. Barwise and J. Seligman, *Information Flow: The Logic of Distributed Systems* (CUP, 1997). Cambridge Tracts in Theoretical Computer Science 44.