Centrum voor Wiskunde en Informatica
REPORT*RAPPORT*

Relational methods in logic, language and information

P. Blackburn, M. de Rijke, and Y. Venema

Computer Science/Department of Software Technology

CS-R9561 1995

Report CS-R9561
ISSN 0169-118X

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

Relational Methods in Logic, Language and Information

Patrick Blackburn[1], Maarten de Rijke[2] and Yde Venema[3]

[1] *Computerlinguistik, Universität des Saarlandes*
Postfach 1150, D-66041 Saarbrücken, Germany
patrick@coli.uni-sb.de
[2] *CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands*
mdr@cwi.nl
[3] *Dept. of Mathematics and Computer Science, Free University*
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
yde@cs.vu.nl

Abstract

This paper discusses the use of relational methods in the interdisciplinary field of Logic, Language and Information. We first sketch the developments that lead up to the current focus on dynamics in the area. After that we give examples of logics of transition that naturally arise in this setting, and we identify more general themes such as bisimulations, relativisations and dynamic modes of inference. We conclude with a discussion of newly emerging themes, and the limitations of the relational perspective.

AMS Subject Classification (1991): 03B40, 03B60, 03B65, 03B80, 03G15, 03G25, 68T30.
CR Subject Classification (1991): F.4.1, I.2.0, I.2.4.
Keywords & Phrases: Relational methods, dynamic logic, natural language, arrow logic, information change.
Note: This report will appear as Chapter 14 of *Relational Methods in Computer Science*, edited by Chris Brink and Gunther Schmidt, in cooperation with Rudolf Albrecht, Springer-Verlag, 1996.

It is difficult to summarise the research that falls under the heading 'Logic, Language and Information' (LLI), for this rapidly evolving interdisciplinary field treats a variety of topics (ranging from knowledge representation to the syntax, semantics and pragmatics of natural language) from a variety of perspectives. However one word more than any other gives the flavour of much contemporary work in LLI: *dynamics*. The purpose of this chapter is twofold. First, we give an impression of what LLI is and why dynamics plays such a fundamental role there. Second, we relate the study of dynamics to relation algebra. The essential point that will emerge is that many LLI approaches to dynamics can be naturally viewed as explorations of *fragments* of relation algebra via their *set theoretic representations*.

We proceed as follows. In the first section we sketch the developments that lead to the current focus on dynamics in LLI. The idea of *logics of transitions* emerges naturally from this discussion, and provides the bridge to the world of relation algebra. In the following section we discuss the syntax and semantics of a number of transitional logics in detail, emphasising the variety of options this essentially simple idea offers. In the third section we turn to more general technical themes. Issues discussed include the key model theoretic notion of a

bisimulation, various metatheoretic properties of these logics and the idea of *relativisation*, and recent work on *dynamic modes of inference*. We conclude with a discussion of newly emerging themes, and the limitations of the relational perspective.

1. Dynamics in logic, language and information

In broad terms, research in LLI aims to give abstract models of high-level information processing. It is reasonably simple to explain what is meant by 'abstract': in principle, it means any mathematical or computational model, though in practice it has tended to mean tools drawn from mathematical logic, theoretical computer science, or the logical and functional programming paradigms. Explaining what is meant by 'high-level information processing' is less straightforward. There is a core intuition that many cognitive abilities such as language understanding, planning and spatial visualisation can be usefully thought of in terms of information processing. Much research in LLI is concerned either to analyse such abilities (usually at a fairly high degree of abstraction) or to develop general models of information and information flow. Emphasis has tended to be placed on those aspects of high-level information processing that readily lend themselves to symbolic analysis, hence the ability of humans to work with beliefs and to cope with language have been the focus of attention. Summing this all up: LLI borders such disciplines as theoretical computer science, linguistics and cognitive science, and its practitioners are theoreticians inspired by problems drawn from these fields.

Stated at such a level of generality, it is probably unsurprising that 'dynamics' has emerged as a key concept. After all, we talk of *forming* beliefs, *amending* them, *changing our mind*, and *learning something new*; the idea implicit in these folk psychological descriptions is of creating, updating or discarding some kind of belief structure. Nonetheless, a little more historical background is necessary to appreciate just why it is that contemporary LLI treatments of dynamics take the form they do — and in particular, why they lead to the study of relational fragments.

Current LLI and its notion of dynamics, has been influenced by three developments: first, the growing realisation (in Linguistics, Computational Linguistics and Artificial Intelligence) that process based explanations at too low a level of abstraction were counterproductive, and the consequent call for more 'logical' or 'declarative' analyses; second, the highly influential insight (in Semantics of Natural language and Belief Revision) that logics equipped with more procedural interpretations could be useful analytic tools; and third, the work in theoretical computer science on logics for reasoning about program behaviour. This last influence has proved particularly important. It has given rise to the idea of general *logics of transitions* or *dynamic logics*, and provides the link with relation algebra. Let us consider these developments in a little more detail.

Early Chomskyan linguistics placed heavy emphasis on procedural explanation: the linguist's task was to explain how grammatical syntactic structures were built up via chains of structure manipulating transformations. By the 1970s the program had degenerated into uninhibited programming over trees that yielded an ever-decreasing return of linguistic insight. Chomsky redirected the field: the task of the 'principles and parameters' approach which emerged was to find the general principles that govern whether or not sentences are well-formed. The question of how the syntactic structure was actually built up now had secondary status.

Over roughly the same period, interest in 'logical approaches' developed in both Compu-

tational Linguists and Artificial Intelligence. Pioneering work on large scale grammars made it clear that processing procedurally specified grammars was difficult. Attention turned to declarative grammar formalisms, in which well-formedness conditions on structure could be described, with little or no procedural commitment; the availability of PROLOG, and the development of unification and constraint solving techniques, ensured that such formalisms also had straightforward procedural realisations. With their combination of declarative clarity and ease of implementation, such approaches swiftly became dominant in computational linguistics. Much the same development occurred in mainstream AI; again, higher level methods which abstracted away from procedural details seemed called for.

In short, many developments in Linguistics, Computational Linguistics and AI can be seen as a move towards logical (or declarative) analyses of problems, that retained a (high-level) procedural content. Although the terminology was not used in these fields, it is natural to sum this up as a quest for dynamic logics. At the same time researchers in Natural Language Semantics and Belief Revision were working in the reverse direction: from logic to computation. Instead of thinking of logic as an essentially static tool, they wanted to infuse it with computational content. Let us consider why.

If one is concerned to give the semantics of a single natural language sentence, then the static truth conditions provided by classical logic (perhaps extended with various modal operators) probably suffice. For example, "A man walks in the park" can be represented as $\exists x\,(\mathit{Man}(x) \wedge \mathit{Walks\text{-}in\text{-}park}(x))$. The standard first-order semantics makes this formula true in a model $\mathcal{M}$ if there is an assignment g of values to variables such that

$$\mathcal{M}, g \models \mathit{Man}(x) \wedge \mathit{Walks\text{-}in\text{-}park}(x).$$

That is, the meaning of this sentence is adequately modeled in terms of the existence of a satisfying assignment. As soon as one considers multi-sentence discourses, however, the limits of classical logic start to show. Consider the following discourse: "A man walks in the park. He whistles." One possible representation of this in classical logic is:

$$\exists x\,(\mathit{Man}(x) \wedge \mathit{Walks\text{-}in\text{-}park}(x) \wedge \mathit{Whistles}(x)).$$

But this 'analysis' assumes too much. The two sentences are presented sequentially, and we understand them in real time; presumably we built up the semantic representation incrementally. However, the natural incremental representation is clearly

$$\exists x\,(\mathit{Man}(x) \wedge \mathit{Walks\text{-}in\text{-}park}(x)) \wedge \mathit{Whistles}(x).$$

This representation is more honest — it reflects sentential structure of the discourse — but it does *not* capture its content. In particular, because the final occurrence of x is free, the anaphoric link between 'He' and 'A man' is lost.

The (now standard) response in LLI is to add a procedural dimension to logical semantics. The meanings of sentences are no longer thought of in terms of static truth conditions. Rather, they are thought of as *context transformers*. A sentence is uttered in a certain context. Its utterance tranforms that context, thus altering the context in which subsequent utterances will be uttered. These successive transformations of context provide the mechanism by which discourse phenomena, such as anaphoric links, are captured. Intuitively, when we utter "A man walks in the park" in some context, we *introduce a new discourse referent* — let us call

it x — and assert that it picks out a man that walks in the park. To put it another way, we 'instruct' our listeners to make a new memory location available, and to associate with this location the information 'is a man' and 'walks'. Given this enriched context, the subsequent utterance of "he walks" makes perfect sense: it can be viewed as an instruction to update the new location with additional information.

In the following section we will consider one way of formalising this idea, viz. dynamic predicate logic; here we'll simply remark that the key idea of adding a procedural dimension to logical semantics underlies other important approaches to Natural Language Semantics, including Discourse Representation Theory (Kamp and Reyle [22]) and File Change Semantics (Heim [19]). The reason for the widespread acceptance of such systems of dynamic semantics is their intuitive appeal combined with their applicability to many important semantic phenomena, such as the interaction of tense and temporal reference. Dynamic semantics has even been successfully applied to problems usually relegated to 'the dustbin of pragmatics'. In particular, it has provided convincing accounts of presupposition; see Beaver [3] and van Eijck [11] for further discussion.

Similar computational metaphors underly LLI treatments of belief revision. For example, Gärdenfors [13] re-examines propositional logic using the idea that a proposition is function taking epistemic states to epistemic states, and Veltman [40]'s update logic gives an 'eliminative' semantics to a uni-modal language (the effect of evaluating formulas is essentially to discard inconsistent information). This view models belief states as something rather like data-bases, and views reasoning about beliefs as the process of adding new information, querying the database, and retracting or replacing information. The Dynamic Modal Logic of van Benthem [5] and de Rijke [34, 37] is a powerful tool for exploring the database metaphor and we examine it in the following section.

We are now ready to make the abstraction that will lead us, via logics of transition, to relation algebra. The dynamic analyses of natural language discourse and belief change sketched above revolve around one central idea: viewing logical interpretation as a process of navigating through a network of states. Rather than thinking of the logical connectives as tools for describing a fixed situation, we view them as instructions which take us from one state to another. We are naturally lead to the idea of *transition systems* (these are the entities we navigate) and various *logics of transition* (corresponding to the various means we choose for moving between states). This view raises many questions that need answering, but let us defer them to the following section and push on to relation algebra.

A transition system is simply a set equipped with a collection of relations (that is, a relational structure). When we fix a choice of connectives we are essentially selecting a number of possibilities for manipulating (or combining) these relations. That is, we are essentially choosing a subset of the combinatoric possibilities offered by relational algebra. Fixing some transition logic corresponds to fixing a reduct of relational algebra.

The way we have approached relation algebra via transition systems should make it clear that we are interested in the set theoretic representations as much, if not more, than the abstract algebras. It is the transition systems that give us the intuitive 'fit' with applications. But of course, once the link with relation algebras is made, transfer of results between the concrete and algebraic domains becomes possible, and fruitful. We will consider such issues later, but first, let us take a closer look at some logics of transition.

2. Logics of transition

By a transition logic we will mean a formalism that is designed to describe transitions. In this section we give an impressionistic sketch of some transition logics that have been considered in the LLI literature. Our starting point will be the mother of all transition logics, propositional dynamic logic (PDL). We then consider various of dimensions of variation. In particular we consider the *choice of connectives*, which leads to a discussion of Dynamic Modal Logic and Lambek Calculus; the issue of *states versus transitions*, which leads to a discussion of arrow logic; and the *nature of states and transitions* which leads to a discussion of Dynamic Predicate Logic. As a final example, we discuss Evolving Algebras.

Propositional dynamic logic. The language of PDL, propositional dynamic logic, (see [32, 18]) contains a modal operator $\langle\alpha\rangle$ for every regular expression α over some set of atomic programs. The intended reading of a formula $\langle\alpha\rangle\varphi$ is that there is an execution of the program α that terminates in a state satisfying φ. PDL expressions are built up from proposition letters p_0, p_1, ..., atomic programs a_0, a_1, ... using the following production rules:

$$\begin{aligned} \varphi &::= p \mid \bot \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\alpha\rangle\varphi \mid [\alpha]\varphi \\ \alpha &::= a \mid \alpha \cup \alpha \mid \alpha\,;\,\alpha \mid \alpha^* \mid \varphi? \end{aligned}$$

The intended reading of the program $\alpha \cup \beta$ is 'do either α or β non-deterministically'; the program $\alpha\,;\,\beta$ stands for 'do α, then do β'; α^* stands for 'iterate α a finite number of times'; and $\varphi?$ is the program that tests for φ and succeeds if φ is true, and fails otherwise. PDL allows one to express various 'safety' and 'liveness' properties of programs:

- $\varphi \rightarrow [\alpha]\psi$: the precondition φ implies the postcondition ψ after every terminating execution of α
- $\langle\alpha\rangle\top$: there exists a terminating execution of α.

Being a modal logic, PDL is interpreted on Kripke structures. Let Prog be the collection of all programs in the language. Then, a Kripke structure for PDL has the form $(W, R_\alpha)_{\alpha\in\mathrm{Prog}}$, where each R_α is a binary relation on W, and

$$\mathcal{M}, w \models \langle\alpha\rangle\varphi \text{ iff for some } v,\ R_\alpha wv \text{ and } \mathcal{M}, v \models \varphi.$$

To reflect the intended readings of PDL's program constructions, we require that our models satisfy the requirements $R_{\alpha\cup\beta} = R_\alpha \cup R_\beta$, $R_{\alpha;\beta} = R_\alpha\,;\,R_\beta$, $R_{\alpha^*} = (R_\alpha)^* = \bigcup_n (R_\alpha)^n$, and $R_{\varphi?} = \{(w,v) \mid w = v \text{ and } \mathcal{M}, w \models \varphi\}$. To get a complete deductive system for PDL, start with the **K** axioms for every modality $\langle\alpha\rangle$ (see Chapter 1 of this volume), and add the following axioms, most of which are decompositions of the program constructions in terms of booleans and simpler programs:

$$\begin{aligned} &\langle\alpha \cup \beta\rangle\varphi \leftrightarrow \langle\alpha\rangle\varphi \vee \langle\beta\rangle\varphi \\ &\langle\alpha\,;\,\beta\rangle\varphi \leftrightarrow \langle\alpha\rangle\langle\beta\rangle\varphi \\ &\langle\varphi?\rangle\psi \leftrightarrow \varphi \wedge \psi \\ &\langle\alpha^*\rangle\varphi \leftrightarrow \varphi \vee \langle\alpha\rangle\langle\alpha^*\rangle\varphi \\ &\varphi \wedge \langle\alpha^*\rangle\neg\varphi \rightarrow \langle\alpha^*\rangle(\varphi \wedge \langle\alpha\rangle\neg\varphi). \end{aligned}$$

The final two axioms are called the Segerberg axioms; they reflect the more complex infinitary behavior of iteration. We refer to Goldblatt [14] for an accessible completeness proof for PDL.

In addition to finite axiomatizability, PDL also enjoys a second important advantage over full relation algebra, namely decidability. To be precise, the satisfiability problem for PDL is EXPTIME-complete (see [12, 33]).

The modal algebras of PDL are two-sorted algebras in the style of the Peirce algebras mentioned in Chapter 1 of this volume. These algebras — called *dynamic algebras* — differ from Peirce algebras in that their relational component is based on a so-called Kleene algebra rather than a relation algebra. A Kleene algebra is a structure $(K, \sqcup, \bot, ;, ^*, \mathbb{I})$, where * is the reflexive, transitive closure operator of Section 1.2 of this volume. We refer the reader to Kozen [24] for the axioms governing the interaction between the two sorts in a dynamic algebra.

Let us briefly list the ingredients that we have at hand now, as most of the transition logics found in the literature may be perceived as variations on PDL. The most important dimensions along which variations have been considered are

- *Connectives.* PDL has the usual booleans to combine formulas the regular operators to build programs, the test operation taking formulas to programs, and the modalities that take a program and a formula to return a formula.
- *Site of evaluation.* The models for PDL have both states and transitions, but PDL formulas are evaluated only at states.
- *The nature of states and transitions.* In PDL no assumptions are made about the nature of the states (although in practical applications they are usually memory structures of some kind), and the transitions are taken to be ordered pairs.

The above dimensions have been varied widely in the literature. The two main (and often conflicting) motivations for these variations are the need for descriptively adequate systems that are able to express all the key features of the phenomena being studied, and the need for computationally well-behaved calculi that have tractable, or at least decidable satisfiability problems. We refer the reader to van Benthem, Muskens, and Visser [9] for an extensive overview; here we will content ourselves with some representative examples.

Choice of connectives. *Dynamic modal logic*, DML, differs from PDL in that its relational component allows all the usual operations from relation algebra in addition to converse operation. Moreover, for every formula φ DML has two special relations: transitions along an abstract information order to states where φ holds ($\mathsf{exp}(\varphi)$), and transitions backward along the information order to states where φ fails ($\mathsf{con}(\varphi)$). On the formula side it has three constructs taking relations to propositions: $\mathsf{dom}(\alpha)$, $\mathsf{ran}(\alpha)$ and $\mathsf{fix}(\alpha)$ are true in a state if it is in the domain, range or set of fix points of α, respectively. All in all, we have

$$\begin{array}{lcl} \varphi & ::= & p \mid \bot \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{dom}(\alpha) \mid \mathsf{ran}(\alpha) \mid \mathsf{fix}(\alpha) \\ \alpha & ::= & \mathsf{con}(\varphi) \mid \mathsf{exp}(\varphi) \mid -\alpha \mid \alpha \cup \alpha \mid \alpha\,;\alpha \mid \alpha^{\smile} \mid \varphi? \end{array}$$

DML is interpreted on *information structures* $(W, \sqsubseteq)$ where W is a non-empty set, and $\sqsubseteq$ is a pre-order on W, called the *information order*; intuitively, $x \sqsubseteq y$ if y is at least as informative as x.

DML expressions are interpreted on information structures by means of a valuation V (to take care of the formulas) and an interpretation $[\![\cdot]\!]$ (to take care of the relational expressions). Some of the clauses in the truth definition are

$$\begin{array}{rcl}
\mathcal{M}, w \models \mathsf{dom}(\alpha) & \text{iff} & \text{there exists } v \text{ with } (w, v) \in [\![\alpha]\!] \\
\mathcal{M}, w \models \mathsf{ran}(\alpha) & \text{iff} & \text{there exists } v \text{ with } (v, w) \in [\![\alpha]\!] \\
\mathcal{M}, w \models \mathsf{fix}(\alpha) & \text{iff} & (w, w) \in [\![\alpha]\!] \\
[\![\mathsf{exp}(\varphi)]\!] & = & \lambda xy.\, (x \sqsubseteq y \wedge y \models \varphi) \\
[\![\mathsf{con}(\varphi)]\!] & = & \lambda xy.\, (x \sqsupseteq y \wedge y \not\models \varphi) \\
[\![\varphi?]\!] & = & \lambda xy.\, (x = y \wedge y \models \varphi).
\end{array}$$

So, the original PDL diamonds can be recovered as $\langle\alpha\rangle\varphi = \mathsf{dom}(\alpha\,;(\varphi?))$, and the dom operator can be expressed in PDL as $\mathsf{dom}(\alpha) = \langle\alpha\rangle\top$.

DML was designed as a general framework for reasoning about information change. Technical results covering expressive power, undecidability and a complete axiomatisation for DML may be found in de Rijke [34]. Uses of DML as a framework for dynamic phenomena may be found in [5, 21, 34]; examples include theory change, update semantics, knowledge representation, and dynamic semantics for natural language.

Site of evaluation. Transitions are commonly depicted as arrows. In a mathematical setting such arrows might denote vectors, functions or morphisms; for a computer scientist they may represent steps in a computation, and to a linguist they may denote the dynamic meaning of a chunk of text. *Arrow logic* is the basic modal logic of arrows. That is, in arrow logic arrows are the sites of evaluation. And so propositions denote sets of arrows, rather than preconditions or postconditions of transitions as in traditional modal logics such as PDL. This doesn't imply that arrows are primitive entities. On the contrary, in the semantics of arrow logic an important role is played by two-dimensional models in which arrows are pairs (a, b) of which a is the start of the arrow, and b is its end.

Following the familiar operations from relation algebra, natural ways of endowing collections of arrows with structure suggest itself. One can think of a ternary relation of *composition* C, where $Cabc$ denotes the fact that the arrow a can be decomposed into two arrows b and c. Or one can designate a subset as the set I of *identity arrows*. The addition of a binary relation R of reverse is slightly more debatable: Rab if the arrow b is a reverse of the arrow a. These ingredients make up an *arrow frame* (W, C, I, R). The *arrow language* is given by the following production rule

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \circ \varphi \mid \otimes\varphi \mid \delta.$$

Here, $\circ$ is interpreted as composition, $\otimes$ as converse, and δ as identity. The language is interpreted on *arrow models* $\mathcal{M} = (\mathcal{F}, V)$ where $\mathcal{F}$ is an arrow frame, and V is a valuation. The interesting clauses in the truth definition are

$$\begin{array}{rcl}
\mathcal{M}, a \models \varphi \circ \psi & \text{iff} & \text{there are } b, c \text{ with } Cabc \text{ and } \mathcal{M}, b \models \varphi \text{ and } \mathcal{M}, c \models \psi \\
\mathcal{M}, a \models \otimes\varphi & \text{iff} & \text{there is } b \text{ with } Rab \text{ and } \mathcal{M}, b \models \varphi \\
\mathcal{M}, a \models \delta & \text{iff} & Ia.
\end{array}$$

The *two-dimensional frames* or *relativized squares* an important and well-known example of an arrow frames. These are built up using a base set U such that the domain W is a subset

of $U \times U$, C satisfies $Cabc$ if $a_0 = b_0$, $a_1 = c_1$, and $b_1 = c_0$; R satisfies Rab iff $a_0 = b_1$ and $a_1 = b_0$; and Ia holds iff $a_0 = a_1$. If W is a reflexive and symmetric binary relation, then the arrow frame is called *locally square*; if W is the full Cartesian square over U, then the frame is called a *square*.

One of the most important connections between arrow logic and relation algebra lies in the fact that relation type algebras are the *complex algebras* of arrow frames. To see some of the benefits of this connection, observe first that, modulo a trivial translation τ (with clauses like $\tau(\varphi \vee \psi) = \tau\varphi + \tau\psi$) arrow formulas are the terms of the algebraic language for relation type algebras. For axiomatic aspects of arrow logic this has the following important consequences. An arrow formula φ is valid on all squares iff $\tau\varphi = 1$ is valid on all representable relation algebras, and φ is valid on all relativized squares iff $\tau\varphi = 1$ is valid on all relativized representable relation algebras. These equivalences form the basis for transferring techniques and results from algebraic logic to arrow logic, and vice versa. We refer the reader to Venema [42] for details and further results.

Axiomatic questions for arrow logic have been studied extensively. For the class of all arrow frames the 'minimal normal arrow logic' (which is the straightforward generalization to the similarity type $\{\circ, \otimes, \delta\}$ of the minimal normal modal logic **K** in the similarity type with just one diamond $\Diamond$ as defined in Chapter 1 of this volume) is complete. For the (relativized) square frames the situation is far more complicated. For instance, the squares are not finitely axiomatizable (this follows from the well-known non-finite axiomatizability result for representable relation algebras).

To conclude our discussion of arrow logic we briefly mention some systems closely related to it. First, the *Lambek Calculus* is a a substructural Gentzen-style derivation system involving a fragment of arrow logic; its connectives are $/$, $\backslash$ and $\circ$, and in relativized square models the slashes are interpreted as the *residuals* of $\circ$. When interpreted on such models, the Lambek Calculus receives a dynamic, procedural interpretation, based on the paradigm that parsing a sentence is performing a logical deduction. A second example concerns *two-dimensional modal logic*; this is an example of a modal system whose intended semantics is based on domains with objects that are tuples over some base set (see Marx and Venema [26] for the general picture). Among others, two-dimensional logics arise in formal analyses of temporal discourse when both a *point of reference* and a *point of utterance* needs to be accounted for. Arrow logic is an example of a two-dimensional logic, as part of its intended semantics is formulated in terms of (relativized) squares.

Finally, there are hybrid systems, languages with two sorts of formulas — one interpreted on states, another interpreted on transitions —, and with a rich set of operators relating the two sorts. Van Benthem [8] presents an abstract approach, Marx [25] has results on concrete interpretations of sorted transition systems, and de Rijke [36] presents a completeness result for the full square case of Peirce algebras.

The nature of states and transitions. To see an example of a calculus in which a very choice is made as to the units that carry semantic information, let us return to the example text in Section 1: "A man walks in the park. He whistles." To formalise the idea explained in Section 1 that meanings of sentences in such a discourse are to be thought of as context transformers rather than in terms of static truth conditions, Groenendijk and Stokhof [15] introduce a system of *dynamic predicate logic* (DPL). Its syntax is the same as that of first-

order logic, and so are its models. What is novel is that it is interpreted with respect to *ordered pairs* of assignments of values to variables; thus, these become the basic units of semantic information. The first assignment is viewed as the 'input'. Evaluating a formula may have the effect of altering this assignment. This new assignment is returned as the second component, the 'output.'

To make thus more concrete, let us give some formal details. First, a model $\mathcal{M}$ is a pair (D, F), where D is a non-empty set, and F is an interpretation function mapping constants and predicates to elements and subsets (of tuples) of the model in the usual way. An assignment is a function assigning individuals to variables; we write $g \approx_x h$ for g and h agree on all variables, except maybe x. Then, given a model $\mathcal{M}$ and an interpretation function F, some of the key clauses in the definition of the semantics for DPL are

$$\begin{aligned} [\![Rt_1 \dots t_n]\!] &= \{(g,h) \mid h = g \text{ and } ([\![t_1]\!] \dots [\![t_n]\!]) \in F(R)\} \\ [\![\varphi \wedge \psi]\!] &= \{(g,h) \mid \exists k\, ((g,k) \in [\![\varphi]\!] \text{ and } (k,h) \in [\![\psi]\!])\} \\ [\![\exists x\, \varphi]\!] &= \{(g,h) \mid \exists k\, (k \approx_x g \text{ and } (k,h) \in [\![\varphi]\!])\}. \end{aligned}$$

So, an atomic statement admits those assignments which satisfy it, and blocks those that don't; conjunction is simply interpreted as composition, and when an existential quantifier is evaluated, a satisfying assignment to the bound variable is recorded.

In the little discourse above, the effect of evaluating $\exists x\, (\mathit{Man}(x) \wedge \mathit{Walks\text{-}in\text{-}park}(x))$ with respect to an input assignment g would be to produce an output assignment f just like g, save that $f(x)$ is a value satisfying $\mathit{Man}(x)$ and $\mathit{Walks\text{-}in\text{-}park}(x)$. This output assignment f is then used as the *input* to the second part of the discourse, $\mathit{Whistles}(x)$. As f fixes a value for x that satisfies $\mathit{Man}(x)$ and $\mathit{Walks\text{-}in\text{-}park}(x)$, the fact that this occurrence of x is free is unproblematic; subsequent interpretation can take place straightforwardly, and the anaphoric link is accounted for.

Interpreting formulas at pairs of assignments instead of single assignments brings with it various choices for the notions of truth and consequence; van Benthem [5] explores a large number of them, and some are discussed in the following section. In addition, the relational perspective can be used to explore the behaviour of DPL's propositional connectives; Blackburn and Venema [10] examines the behaviour of dynamic implication in both the presence and absence of the boolean operations. Finally, we should mention that there are close similarities between quantified versions of the logic PDL discussed above and DPL; Groenendijk and Stokhof [15] study various embeddings of the latter into the former.

Anything goes. To conclude our impressionistic tour of transition logics, we briefly discuss *evolving algebras*. The subject of evolving algebras was introduced by Yuri Gurevich around 1985 as a means of specifying operational aspects of computation at a level appropriate for reasoning about the system in question. Evolving algebras can be viewed as vast generalization of PDL (see [17]). It uses many-sorted first-order structures as the *states* of its structures, and in contrast to standard practice, the constant and function symbols are dynamic: they might change according to a set of *transition rules*. Although we can't give any formal details here, the evolving algebra framework seems to provide some general tools for useful specification, and the recent literature indicates that these tools might be fruitfully applied in LLI as well (see for example Moss and Johnson [29]).

3. General Themes

The aim of this section is to acquaint the reader with the kind of questions that the dynamic perspective on semantics brings to the fore. Far from being able to give an comprehensive overview in these few pages, we decided to highlight a number of ideas and results that we like. These fall under three headings; first, we discuss a connection between relational algebra and the important notion of a bisimulation between transition systems; second, we show a few ways to overcome the negative properties of the relation algebraic approach; and finally, we discuss the notion of semantic consequence in the dynamic way of thinking.

Bisimulations. The simplest models for dynamics are the (labeled) transition systems that we encountered in the previous sections; let us define here, for a given set A of atomic actions or programs, a *transition system* for A as a structure $\mathcal{X} = (X, R_a)_{a \in A}$, where each R_a is a binary relation on X. Such a transition system is supposed to model a process; a relation R_a holding of two states s and t signifies the possibility that an action a is performed in s and leads to t. In many applications, for instance in the semantics of process algebra, cf. Baeten and Weijland [2], a transition system has a designated state called its *root*, which is supposed to represent the initial state of the process.

An important question is now, when two transition systems represent the same process. In other words, we are looking for a natural notion of equivalence between transition systems. In the literature on process theory, one sees many different options. For instance, one might be interested only in the various *traces* of a rooted transition system, i.e., sequences consisting of atomic actions, corresponding to the paths one can take through the transition system (starting from its root). In this approach, two transition systems are equivalent iff they generate the same set of traces. We will concentrate here on a different notion of equivalence between processes here, viz. that of a bisimulation. When compared to trace equivalence, the crucial point in the definition of a bisimulation is, that two states are bisimilar only if one has precisely the same choice of actions enabled in each state.

Formally, a *bisimulation* between two transition systems $\mathcal{X} = (X, R_a)_{a \in A}$ and $\mathcal{X}' = (X', R'_a)_{a \in A}$ is a non-empty relation $Z \subseteq X \times X'$ that satisfies the following *back* and *forth* conditions:

(*forth*) xR_ay and xZx' imply the existence of a y' such that $x'R'_ay'$ and yZy',

(*back*) $x'R'_ay'$ and xZx' imply the existence of a y such that xR_ay and yZy'.

There is another reason why bisimulations have received attention and that is their intimate relation with modal logics. As it was put into a slogan in de Rijke's dissertation [35]: bisimulations are to modal logic what partial isomorphisms are to first-order logic. For instance, it is quite easy to prove that bisimilar states satisfy the same poly-modal formulas (in a language with a modality $\langle a \rangle$ for each action a). A deeper result, due to van Benthem [4], characterizes the 'modal fragment' of first-order logic as the set of those formulas that are invariant under bisimulations.

Then, relational *algebra* may come into the picture in a number of ways; we have to confine ourselves to the following example. Suppose that Z is a bisimulation with respect to the transition systems (X, R, S) and (X', R', S'). It is easy to show that Z is also a bisimulation for the union of the relations, i.e., between the systems $(X, R \cup S)$, $(X', R' \cup S')$, or for the

(reflexive) transitive closure of one of them, e.g., between (X, R^+) and (X', R'^+). On the other hand, one can give an easy counterexample showing that Z does not always have to be a bisimulation with respect to the intersections $R \cap S$ and $R' \cap S'$. In other words, the following definition is meaningful.

A relational operation $O(R_1, \ldots, R_n)$ is *safe for bisimulation* if a relation Z is a bisimulation between the structures $(X, O(R_1, \ldots, R_n))$ and $(X', O(R'_1, \ldots, R'_n))$ whenever Z is a bisimulation between $(X, R_1, \ldots, R_n)$ and $(X', R'_1, \ldots, R'_n)$. Apart from the aforementioned union and (reflexive) transitive closure, examples of safe operations for bisimulation include the diagonal relation (i.e., seen as a constant operation), relation composition and dynamic negation '$\sim$', given by

$$\sim R = \{(x, y) : \text{ not } \exists y\, (xRy)\}$$

In fact, among the relational operations that can be defined in first-order logic, these operations generate the clone of safe operations for bisimulations, as the following theorem shows (for a proof we refer to van Benthem [7]).

Theorem 3.1 *A first-order definable relational operation $O(R_1, \ldots, R_n)$ is safe for bisimulation if and only if it can be defined from the atomic relations R_i and the diagonal relation, using just the three operations ;, $\cup$ and $\sim$.*

The notion of a bisimulation was introduced by van Benthem in [4] (under the name of a p-relation); in the literature on process theory, bisimulations go back to Park [30]. Further references can be found in Ponse, de Rijke and Venema [31].

Taming logics. In earlier chapters we saw that the standard Tarskian framework of relation algebras, although very elegant from the mathematical perspective, has two properties that make it less attractive from the applicational point of view: compared to for instance boolean algebras, the equational theory of the class RRA of representable relation algebras does not have a 'clean' equational axiomatization, and it is highly undecidable as well. A number of results recently obtained in the field of LLI, can be grouped together, under the common denominator that the authors all try to fine-tune or modify the standard approach to get around these negative results. The aim is, to use a slogan of Mikulás [27], how to *tame* transition logics; here, we will mention a few ideas and results that have arisen in the (recent) literature[1]. The various modifications that can tame a logic's undecidability can be divided into the following three areas: studying *reducts* of the original language, interpreting the language in so-called *non-square* models — this is the modal-logic counterpart of *relativizing* relation algebras, or *restricting* the admissible valuations on the full square models.

To start with the second approach, recall that the class RRA is generated by the full relation algebras, and that the full relation algebra on a set U is based on the power set of the *full* cartesian square U^2 over U. In other words, the assumption is that all elements of U^2 are available as transitions, or, possible worlds (in the arrow logic perspective). In the non-square or relativized approach, this assumption is dropped: any subset W of U^2 can serve as the

[1]Our presentation is not justified from the historical perspective. For instance, the idea of a relational interpretation of Lambek's Calculus is due to van Benthem [5], 30 years after the introduction of the system by Lambek.

top set of the algebra. Formally, the *W-relativized full relation algebra* on U is defined as the structure

$$Re^W(\mathsf{U}) = (\mathcal{P}(W), \cup^W, {}^{-W}, \emptyset, ;, {}^{\breve{}W}, \mathrm{I}^W_{\mathsf{U}}),$$

i.e., all relational algebraic operations are *relativized to* W. For instance, the operation ; is given by

$$R ; S = (R ; S) \cap W.$$

Obviously, the equational theory of such non-square algebras is weaker than that of the square ones; for instance, the operation $(\cdot)^{\breve{}W}$ is not idempotent unless W is symmetric.

Now an interesting landscape of algebras arises if we start from various classes of algebras $Re^W(\mathsf{U})$ for which W meets some constraints. Formally, let R stand for reflexive, S for symmetric and T for transitive; let H be a subset of $\{R, S, T\}$ and W a binary relation. We use "W is an H-relation" to abbreviate that W has the properties mentioned in H. We define an algebra to be in RRA_H if it can be embedded in an algebra of the form $Re^W(\mathsf{U})$, with W an H-relation. Then, the effect of taming RRA can be summarized by the following theorem:

Theorem 3.2 *For every $H \subseteq \{R, S, T\}$, the class RRA_H is a variety. This variety is finitely axiomatizable and decidable if and only if $T \notin H$.*

For lack of space, we cannot give proper references to all results covered by this theorem; for a more extensive overview and references, the reader can consult Marx and Venema [26]. The proof that RRA_{RST} (which happens to be the same variety as RRA) cannot finitely axiomatized is due to Monk, the other negative results are due to Andréka, Németi and Sain. Positive results concerning finite axiomatizability are due to Maddux (namely, that RRA_{RS} is identical to the equationally defined variety WA of so-called weakly associative relation algebras), Kramer ($\mathsf{RRA}_\emptyset$) and Marx (the remaining varieties). Concerning computational properties, the undecidability results are due to Tarski (RRA), and Andréka and Németi (RRA_H with $T \in H$), the positive ones to Németi (RRA_{RS}) and Marx (the remaining ones).

The upshot of Theorem 3.2 is that *transitivity* of the top element W is the malefactor. It may therefore come as a surprise that by restricting the language of arrow logic (or algebraically, considering reducts of the relation algebras), there is the following positive result. Theorem 2.1 of Andréka and Mikulás [1] states that the original Lambek Calculus (mentioned in the previous section) is sound and complete with respect to a a relativized relational interpretation with a transitive top set. This result is further strengthened to obtain an answer to an old question posed in Schein [38], viz. to give an axiomatic characterisation of the class of algebras isomorphic to sets of binary relations with the operations of multiplication and two residuations.

A different 'taming strategy', originating with two-dimensional temporal logic, is the following. Let $<$ be a designated ordering relation on the base set U, i.e., in the algebraic language, there is a special constant referring to this relation, just like the 0 always refers to the empty relation. Now consider the subalgebra of $Re(\mathsf{U})$ that is generated by $<$ and arbitrarily many left-ideal elements (essentially, unary relations in disguise). It follows from results in Venema [41], that the set of equations valid in such algebras is decidable, and finitely axiomatizable if we consider algebras only in which $<$ is a well-ordering or the ordering of the natural numbers.

Finally, if one is only interested in axiomatizations of the equational theory of the full relation algebras, the only solution to Monk's negative result on finite axiomatizability is to be more liberal concerning the format of a derivation system. Results in Marx and Venema [26] and Mikulás [27] show that RRA can be finitely axiomatized if one allows *unorthodox derivation rules*.

Dynamic consequence. As pointed out in Section 2, whenever we turn the meaning of a formula into a dynamic notion (in this contribution represented as a set of transitions, hence, as a binary relation), the question arises as to what the proper notion of semantic *consequence* is. In other words, what does it mean that a formula is a consequence of some (finite) bunch of other formulas, in this new semantical perspective? Or, more formally, if φ and $\varphi_1, \ldots, \varphi_n$ are formulas, what is an appropriate definition of the dynamic consequence relation $\models_d$ in (3.1) below?

$$\varphi_1, \ldots, \varphi_n \models_d \varphi. \tag{3.1}$$

To see the point of this question, the reader could try to look at the φ_i's as consecutive updates; it will then become clear that the *order* of the premisses $\varphi_1, \ldots, \varphi_n$ may influence the outcome of the semantic consequence. In other words, the 'static' interpretation

$$[\![\varphi_1]\!] \cap \ldots \cap [\![\varphi_n]\!] \subseteq [\![\varphi]\!] \tag{3.2}$$

does not seem the most appropriate choice for (3.1). In the sequel, we will briefly discuss two dynamic alternatives to (3.2). The relevance of relational algebra lies in the fact, that these interpretations can all be expressed in a quite simple fragment of the language of Tarksi's relation algebras, and, hence, that meta-theoretic notions such as dynamic consequence can be studied at the object-level using relational methods.

In the *dynamic* style of inference we interpret (3.1) by saying that in all models, each transition for the sequential composition of the premises must be admissible for the conclusion:

$$[\![\varphi_1]\!]; \ldots; [\![\varphi_n]\!] \subseteq [\![\varphi]\!] \tag{3.3}$$

In a formal framework for update semantics, the natural interpretation for (3.1) seems to be the following *update* or *mixed* style of inference, "first process all premises consecutively, then test if the conclusion is satisfied by the remaining state":

$$\mathsf{ran}([\![\varphi_1]\!]; \ldots; [\![\varphi_n]\!]) \subseteq \mathsf{fix}[\![\varphi]\!] \tag{3.4}$$

where $x \in \mathsf{fix}R$ iff $(x, x) \in R$ as in Section 2.

Obviously, these new styles of inference do not satisfy all standard structural properties of static inference, such as:

(*Monotonicity*) $X, Z, Y \models \varphi$ if $X, Y \models \varphi$

(*Reflexivity*) $\varphi \models \varphi$

(*Cut*) $Y, X, Z \models \varphi$ if $X \models \psi$ and $Y, \psi, Z \models \varphi$.

(Here X,Y and Z stand for arbitrary sequences of formulas.) For instance, the dynamic inference style (3.3) does not satisfy (Monotonicity), while the mixed style (3.4) satisfies none of the three mentioned properties. However, there are modified versions of these rules which the mixed style does satisfy.

Finally, it is shown by van Benthem that various styles of inference are *characterized*, in some sense, by the structural rules that they admit. For instance, the dynamic interpretation (3.3) is completely determined by the rules (Reflexivity) and (Cut). For reasons of space limitations we cannot go into detail; let us just mention here that these results amount to finite axiomatizations of various very simple *generalized reducts* of representable relation algebras. The interested reader is referred to van Benthem [6], or to Kanazawa [23] or Groeneveld and Veltman [16] for more recent results.

4. Conclusions and New Directions

In this chapter we have tried to give the reader a taste of the use of relational methods in Logic, Language and Information, both by given specific examples of logics of transition and their motivation, and by identifying some general themes. As a result of the interaction between theory and application, the use of relational methods in LLI is undergoing rapid changes; in some cases these changes lead to a more intimate connection between relational methods and LLI, in other cases LLI seems to move away from relational methods. To conclude the chapter we sketch examples of both phenomena.

One of the classical themes in relation algebra, and indeed in algebraic logic, has been the study of restricted or finite variable fragments of first-order logic (see Tarski and Givant [39]). As we have seen in Section 2, the use of relational methods in LLI gives rise to new ways of interpreting well-known systems such as first-order logic. Procedural interpretations of the latter turn out to have an unexpected impact on its expressive power: more can be said with fewer variables, as becomes clear from a detailed case study by Hollenberg and Vermeulen [20]. Their work shows how relational methods give us the tools for dealing with familiar issues of expressive power in the setting of interesting new logics arising in LLI, and how such logics 'give rise to a refreshing view' on those issues.

A natural question at this point is: how far do relational methods get us in understanding dynamic phenomena in LLI? At this stage it is impossible to answer the question, but it is clear that several researchers feel the limitations of relational methods. For example, in their search for a precise specification of the dynamic interpretation process of natural language texts in humans and machines, Visser and Vermeulen [43] take the radical position that one must be able to interpret any chunk of text, and that the interpretation of larger chunks is a function of the interpretations of the smaller chunks. They argue that category theory is the proper format for the description of the flow of interpretation, and for studying the monoidal manner in which interpretations of small chunks of text interact with each other. In a similar vein, Moshier [28] uses category theory to provide a better meta-theory for investigating independence of syntactic principles and enforcement of interactions between principles. And, finally, the work by Moss and Johnson [29] cited in Section 2 on using evolving algebras for analysing grammar formalisms is another example of research in LLI that goes beyond relational methods. To conclude, then, it is not clear how far relational methods will take us, but their limitations are being felt by a number of researchers.

Acknowledgment. The second author was supported by the Netherlands Organization for Scientific Research (NWO), project NF 102/62-356 'Structural and Semantic Parallels in Natural Languages and Programming Languages'.

The research of the third author has been made possible by a fellowship of the Royal Netherlands Academy of Arts and Sciences.

References

1. H. Andréka and S. Mikulás. Lambek calculus and its relational semantics: Completeness and incompleteness. *Journal of Logic, Language and Information*, 3(1):1–38, 1994.
2. J.C.M. Baeten and W.P. Weijland. *Process Algebra*, volume 18 of *Tracts in Theoretical Computer Science*. Cambridge University Press, 1990.
3. D. Beaver. *Presupposition and Assertion in Dynamic Semantics*. PhD thesis, University of Edinburgh, 1995.
4. J. van Benthem. *Modal Correspondence Theory*. PhD thesis, Mathematisch Instituut & Instituut voor Grondslagenonderzoek, University of Amsterdam, 1976.
5. J. van Benthem. Semantic parallels in natural language and computation. In M. Garrido, editor, *Logic Colloquium 1988*. North-Holland, Amsterdam, 1989.
6. J. van Benthem. Logic and the flow of information. In D. Prawitz, B. Skyrms, and D. Westerståhl, editors, *Proceedings 9th International Congress of Logic, Methodology and Philosophy of Science, Uppsala 1991*, pages 693–724, Amsterdam, 1993. Elsevier Science Publishers.
7. J. van Benthem. Programming operations that are safe for bisimulations. CSLI Research Report 93-197, CSLI, Stanford University, 1993. To appear in *Logic Colloquium 1994*, North-Holland, Amsterdam.
8. J. van Benthem. Dynamic arrow logic. In J. van Eijck and A. Visser, editors, *Logic and Information Flow*. MIT Press, Cambridge, Mass., 1994.
9. J. van Benthem, R. Muskens, and A. Visser. Dynamics. In J. van Benthem and A. ter Meulen, editors, *Handbook of Logic and Language*. Elsevier Science, Amsterdam, to appear.
10. P. Blackburn and Y. Venema. Dynamic squares. Logic Group Preprint Series No. 92, Department of Philosophy, Utrecht University, 1993. To appear in *Journal of Philosphical Logic*.
11. J. van Eijck. Presupposition failure — a comedy of errors. *Formal Aspects of Computing*, 6A:766–787, 1994.
12. M. Fischer and R. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18:194–211, 1979.
13. P. Gärdenfors. *Knowledge in Flux*. MIT Press, Cambridge, Mass., 1988.
14. R. Goldblatt. *Logics of Time and Computation*. CSLI Publications, Stanford, 1987.
15. J. Groenendijk and M. Stokhof. Dynamic predicate logic. *Linguistics and Philosophy*, 14:39–100, 1991.

16. W. Groeneveld and F. Veltman. Inference systems for update semantics. Manuscript, ILLC, Amsterdam, 1994.

17. Y. Gurevich. Evolving algebras: A tutorial introduction. *Bulletin of the European Association for Theoretical Computer Science*, 43:264–286, 1991.

18. D. Harel. Dynamic logic. In D.M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic, Vol. II*, pages 497–604. Reidel, Dordrecht, 1984.

19. I. Heim. File change semantics and the familiarity theory of definites. In R. Bäuerle, C. Schwarze, and A. Von Stechow, editors, *Meaning, Use and Interpretation of Language*. De Gruyter, Berlin, 1983.

20. M. Hollenberg and K. Vermeulen. Counting variables in a dynamic setting. Technical report, Department of Philosophy, Utrecht University, 1994.

21. J. Jaspars and E. Krahmer. Unified dynamics. Technical Report CS-R95, CWI, Amsterdam, to appear.

22. H. Kamp and U. Reyle. *From Discourse to Logic*. Kluwer Academic Publisher, Dordrecht, 1993.

23. M. Kanazawa. Completeness and decidability of the mixed style of inference with composition. In P. Dekker and M. Stokhof, editors, *Proceedings of the Ninth Amsterdam Colloquium*, pages 377–391, Amsterdam, 1994. ILLC.

24. D. Kozen. On the duality of dynamic algebras and Kripke models. In *Logic of Programs 1981*, LNCS 651, pages 1–11, Berlin, 1981. Springer-Verlag.

25. M. Marx. Dynamic arrow logic with pairs. In M. Marx and L. Polos, editors, *Arrow Logic and Multi-Modal Logic*, Studies in Logic, Language and Information. CSLI Publications, Stanford, to appear.

26. M. Marx and Y. Venema. *Multi-Dimensional Modal Logic*. Kluwer Academic Press, to appear.

27. S. Mikulás. *Taming Logics*. PhD thesis, ILLC Dissertation Series 1995–12, 1995.

28. M.A. Moshier. Featureless HPSG, 1995. Unpublished manuscript.

29. L.S. Moss and D.E. Johnson. Dynamic interpretations of constraint-based grammar formalisms. *Journal of Logic, Language and Information*, 4:61–79, 1995.

30. D. Park. Concurrency and automata on infinite sequences. In *Proceedings 5th GI Conference*, pages 167–183, New York, 1981. Springer.

31. A Ponse, M. de Rijke, and Y. Venema, editors. *Modal Logic and Process Algebra*, number 53 in CSLI Lecture Notes, Stanford, 1995. CSLI Publications.

32. V. Pratt. Semantical considerations on Floyd-Hoare logic. In *Proc. 17th IEEE Symp. on Foundations of Computer Science*, pages 109–121, 1976.

33. V. Pratt. Models of program logics. In *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*, pages 115–122, 1979.

34. M. de Rijke. A system of dynamic modal logic. CSLI Research Report 92-170, Stanford University, 1992. To appear in *Journal of Philosophical Logic*.

35. M. de Rijke. *Extending Modal Logic*. PhD thesis, ILLC Dissertation series 1993-4, 1993.

36. M. de Rijke. The logic of Peirce algebras. Technical Report CS-R9467, CWI, Amsterdam, 1994. To appear in *Journal of Logic, Language and Information*.

37. M. de Rijke. Meeting some neighbours. In J. van Eijck and A. Visser, editors, *Logic and Information Flow*, pages 170–195. MIT Press, Cambridge, Mass., 1994.

38. B. Schein. Relation algebras and function semigroups. *Semigroup Forum*, 1(1):1–61, 1970.

39. A. Tarski and S. Givant. *A Formalization of Set Theory without Variables*, volume 41 of *Colloquium Publications*. AMS, 1987.

40. F. Veltman. Defaults in update semantics. *Journal of Philosophical Logic*, to appear.

41. Y. Venema. Completeness through flatness. In D.M. Gabbay and H.J. Ohlbach, editors, *Temporal Logic, First International Conference, ICTL'94*, number 827 in LNCS, pages 149–164, Berlin, 1994. Springer.

42. Y. Venema. A crash course in arrow logic. In M. Marx and L. Polos, editors, *Arrow Logic and Multi-Modal Logic*, Studies in Logic, Language and Information. CSLI Publications, Stanford, 1995.

43. A. Visser and K. Vermeulen. Dynamic bracketing and discourse representation. Technical report, Department of Philosophy, Utrecht University, 1995.