DYNAMIC ODDS & ENDS

Johan van Benthem

ILLC, University of Amsterdam
July 1998

This is a bunch of papers devoted to various aspects of logical dynamics.
It is the 1998 sequel to last year's collection "Dynamic Bits and Pieces",
which appeared as ILLC Research Report LP–97–01.

Contents

1 Exploring Logical Dynamics: The Main Lines

"Exploring Logical Dynamics" consists of three, not tightly connected parts: a survey of current trends in dynamic semantics (ch's 1–2), a process theory in extended modal logic (ch's 4–9), a bunch of illustrations of these phenomena in various fields (ch's ≥10).This introduction sums up what I see as the main points of the ELD monograph.

1 The proposed paradigm: modal logics of process graphs

The core of the book are Chapters 4–9, which propose a mathematical paradigm for the Dynamic Turn. Process theories can be designed as <u>modal logics of process graphs</u>, not via one unique system, but as a family varying in expressive strength and deductive power. Thus, between the lines, the book also presents 'modal logic in a new key' – with its repercussions for standard parts of logic. In other words, the book basically proposes a *general methodology* for the analysis and design of dynamic systems, with theorems backing up its viability and interest.

2 Three main methods: bisimulation, guards, and correspondence

Now, the question arises at once how one avoids a steaming jungle of new systems. What provides uniformity in the theory? The answer lies in two general viewpoints and techniques, which form an independent contribution of ELD, beyond 'dynamics'. The main innovations are (1) systematic use of <u>bisimulations</u> (in a broad sense), which allows for a model theory on classical lines, (2) syntactic <u>guarded quantifier</u> analysis, as a 'thermometer' for expressive power and computational complexity. A third red thread through the book are (3) modal <u>frame correspondences</u> (cf. my 1985 monograph "Modal Logic and Classical Logic"), which identify the computational import of special axioms on top of the minimal logic for dynamic languages.

3 The tandem approach: both modal and classical

Typical for ELD is a duality between 'modal' and 'standard' viewpoints. These are two sides of the same coin, modulo effective translation. (In particular, there is no need for *choosing* between the two, as some puritans think.) This style of working again has more general import. It allows us to use insights from standard logic in the new dynamic logic, instead of setting up the cottage industries that make so much of computer science disconnected. It also suggests new process logics that would not easily come up otherwise. A typical example are the new logics for parallellism in LICS 98 (Section 2) – not discussed in ELD, which concentrates on sequential actions. First, one finds more delicate simulations involved with operations for 'joint action'. Next, guarded analysis shows how matching languages skirt the edge of undecidability.

4 The propositional core logic of dynamics: main themes, and repercussions

The ELD framework is an abstract 'propositional logic' of dynamics. This level is very poor. All the greater the success if one finds significant questions here! By my count, these are four.

(**1**) General theory of semantic simulations and matching syntactic expressive power, with the 1976 bisimulation invariance theorem as a point of departure (Ch. 4). (**2**) Analysis of 'natural' process operations (Ch. 5). Practising dynamic semanticists find this concern abstruse (some computer scientist find it the outstanding question of their field). Noone knows how to address this well. It is akin to the vexed question what are 'logical constants'. The ELD proposal reads: 'safety for (bi-)simulation', strengthening Tarski's 'permutation invariance'. The key result is the 1993 safety theorem, which cuts down the first-order operations to essentially obvious dynamic readings of negation, conjunction and disjunction – and thus identifies a natural 'propositional core' for dynamics. In ELD methodology, this is not the end: as we seek generality across many kinds of simulation. It is a pilot for a type of expressive completeness result which I would like to put on the agenda. (**3**) Interplay between expressive power/computational complexity for dynamic logics. Here the picture reverses. We use a dynamic perspective to take a new look at standard logic, finding large decidable fragments. Main result: decidability of the Guarded Fragment, as a pilot for other systems in Chapters 4, 9. (**4**) Modal/dynamic reinterpretation of standard logic (Chapters 8, 9). The general issue now is identification of 'hidden parameters' in standard modeling: 'dependence' is a prime example. The outcome is a new perspective on standard logic, which might change its teaching. Frame correspondences determine surprising computational content for formerly 'anonymous' standard laws of first-order logic. Over our generalized modal semantics, various dynamic extensions for the classical language emerge.

5 Striking omissions

(**1**) No systematic analysis of complexity for decidable dynamic logics: deeper fine-structure remains unexplored. (**2**) No fixed-point versions of systems, so that we miss operations crucial to real computation. (**3**) No systematic exploration of additional axioms on top of the minimal logics, as in standard modal logic. (**4**) No analysis of parallel or 'joint' action. Of course, these are all obvious next agenda items – and we know more now, two years after ELD's appearance.

6 Relating other dynamic approaches

ELD proposes and develops a modal paradigm. It does not say that *other* approaches to dynamics are wrong (linear logic, game theory, process algebra, &c). What I *would* claim, two years later, is an additional virtue. One can often profitably *analyze* other approaches in the ELD style. A good example is the modally inspired analysis of Chu Spaces (Section 6), and another the modal analysis of *game logics* – which will be the subject of a later ILLC report.

2 Process Operations in Extended Dynamic Logic

This is an extended abstract for a tutorial at "Logic in Computer Science", LICS 98, Indianapolis, which was delivered eventually by Maarten de Rijke(whose slides with additional material can be obtained via email mdr@wins.uva.nl). The text outlines the main program of guarded first-order analysis for process theories. Further clarifications of definitions and results are in Section 3.

Abstract

Modal logic becomes action logic by adding programs as in propositional dynamic logic or the μ–calculus. Modal languages can be seen as decidable fragments of first-order logic that admit a natural bisimulation, and hence enjoy a good model theory. Recently, much stronger 'guarded fragments' of first-order logic have been identified that enjoy the same pleasant features. The latter can serve as richer action languages as well. We will develop the logic of guarded fragments as a form of process theory. ln particular, moving from sequential to parallel process operations correlates with moving to first-order fragments that are close to, or perhaps just over the decidable–undecidable fence.

1 The modal dynamics of actions

We will start by reviewing the basics. Standard poly-modal logic is a decidable fragment of the first-order logic of process graphs (labeled transition systems, Kripke models). It can be characterized semantically as consisting, up to logical equivalence, of those first-order formulas which are invariant for *bisimulation*.

Propositional dynamic logic turns this into an explicit action language by treating propositions and programs on a par, adding a syntactic component of regular programs, including tests for all propositions. Again, this system is decidable, its propositions are invariant for bisimulation, while its programs are what may be called 'safe for bisimulation'. (Roughly speaking, transition relations for all programs enjoy automatic zig-zag over any existing bisimulation).

To obtain the full power of fixed-point operations over all syntactically positive predicate transformers, however, one must move to the modal μ-calculus. Again, the latter system is decidable, and it consists of all bisimulation-invariant statements in a first-order logic with fixed-point operators over process graphs. (This convenient paraphrase of a recent semantic characterization is equivalent to the version involving monadic second-order logic.)

This line of logics runs into clear limitations, as it does not handle joint or parallel action. But read on.

2 From modal to guarded logics

Modal logic behaves much like a miniature of first-order logic in its main system properties (effective axiomatizability, interpolation, preservation results). The mechanism that drives this strong similarity is essentially the following meta-equation:

ML : FOL = *bisimulation* : *potential isomorphism*

We will unpack this terse, but meaning-laden statement somewhat in the tutorial. Of course, modal logic achieves all this while staying decidable. Recently, it has become clear these virtues are shared by much larger decidable parts of first-order logic. A typical example is the *Guarded Fragment* (GF), allowing all existential quantifications of the form

$$\exists \mathbf{y}\ (G\ (\mathbf{x}, \mathbf{y})\ \&\ \phi\ (\mathbf{x}, \mathbf{y}))$$

Here $\mathbf{x}, \mathbf{y}$ are finite sequences of variables, and the 'guard' G ($\mathbf{x}, \mathbf{y}$) is an atom in which these variables all occur, in any order or multiplicity of occurrence (Rxy, Ryx, Rxyx, etc.). Also, GF has no restriction to specifically designated predicates for guards – like the special relational guard 'R' found in modal logic. The matrix statement ϕ is again a guarded formula. GF admits of a natural bisimulation analysis, and it is decidable (complete for doubly-exponential time).

With designated guard predicates, one gets the weaker but useful *action-guarded fragment* A-GF, which makes a principled distinction between state predicates and action predicates. A-GF enjoys the same properties as GF (its natural decidability proof is even somewhat more 'constructive'). Moreover, both GF and A-GF have a standard model theory.

First-order translation from modal languages into GF explains many known scattered decidability results (minimal modal and tense logic, additional frame conditions). A current focus are decidable extensions, explaining even more. For instance, decidability of Since/Until temporal logic reduces to decidability of GF extended with guards that are atomic conjunctions which are 'pairwise guarded': that is, any two variables from $\mathbf{x}, \mathbf{y}$ occur together in at least one guard atom.

The location of the 'undecidability threshold' for full predicate logic is a subtle matter here. Allowing (1) matrix statements introducing new free variables, or (2) arbitrary conjunctions in guards, leads to undecidable languages. Guards provide a new take on decidable fragments of FOL, different from the usual divisions (arities, prenex forms, finite variable sets). They are rather related to general algebraic techniques of 'relativization' for various undecidable logics.

Another way of pushing the threshold moves beyond first-order logic to *fixed-point extensions* of guarded fragments. For instance, while transitivity of relations is non-guarded (and bad for decidability...), the well-known decidability of modal S4 on transitive models may be explained by translation into a fixed-point extension of GF, generalizing the μ-calculus in an obvious way. Decidability of these (modestly non-first-order) systems remains a conjecture at present.

The tutorial will cover the basic theory of these guarded fragments, as compared with full FOL.

3 Connection with process logics

Guarded languages evidently provide richer process representations than standard modal ones. They allow for complex states (through the use of tuples), and thereby to more complex transitions between these. Our main theme in this tutorial is loosely described by the following general 'meta-equation':

$$\text{PDL} : \mathbf{?} = \text{ML} : \text{GF}$$

That is, how can we strengthen PDL to achieve the benefits that the guarded language offers over standard modal logic? Read in another way, of course, we have

$$\text{GF} : \mathbf{?} = \text{ML} : \text{PDL}$$

What is a good action view of guarded languages? We can extend both questions to include fixed points.

The main aim of the tutorial is to demonstrate how one can usefully think of process languages and decidable fragments of standard logics in tandem. For instance, the action-guarded fragment A-GF talks about transitions between complex states where the only evaluations that we make concern those states. (My complaining about the noise changes the state from one with a defective fan to one with a good fan.) GF allows also comparisons across these states. (My complaining made me happier now than I was then.) In parallel action, we would also wish to decompose what happens to components of the state, and hence have non-atomic, conjunctive guards. (You complain about the pump, and I'll deal with the fan - and who knows, I'll be happier now than you were then.) The pairwise guarded fragment urges us to state all cross-comparisons between effects from input to output.

Actually, in this interplay, the difference between special 'modal' or 'dynamic' formalisms and their first-order guarded counterparts becomes slight. So there is a real issue (familiar from other areas of applied logic) why we could not use suitable fragments of first-order logic directly, rather than go for new language design.

4 Sequential action on multi-states

4.1 Joint Action over State Tuples

Collective states may have many components. This can be represented by moving from binary transition relations to general *finitary relations* $R\mathbf{xy}$ between finite sequences of individual states. One language for this is a many-dimensional modal one, with two components: state predicates, and action predicates. This requires a two-level syntax, as for PDL, plus some book-keeping of arities for both levels (position numbers, or with variables themselves as 'positions'). We will discuss this use of variables in the tutorial. We outline the main notions and results, skipping the technicalities of formal notations or proofs.

Assertions. State atoms $P\mathbf{x}$, all Boolean operations, existential modal operators $\langle R\rangle_{\mathbf{x},\mathbf{y}}$ (taking $\mathbf{y}$-state formulas to $\mathbf{x}$-state formulas) and 'lifters' $[\phi, T]_{\mathbf{z}}$ (from $\mathbf{x}$-state formulas ϕ to $\mathbf{x+z}$-state ones).

Programs. Action atoms $R_{\mathbf{x},\mathbf{y}}$, relation composition (with arity fit), union (with arity fit), tests $(\phi)\mathbf{?}$, projections $\Pi_{\mathbf{x},\mathbf{y}}$ (from a larger $\mathbf{x}$ to a subset $\mathbf{y}$).

Models and a truth definition for this language are like in many-dimensional modal languages. In particular, the lifter holds at an $\mathbf{x+z}$-tuple if ϕ holds at its $\mathbf{x}$-subtuple. That the above is a natural set of operators shows, amongst others, in the later 'safety analysis'.

There exists a straightforward *effective translation* from this system taking both assertions and programs to formulas of the action-guarded fragment A-GF. (This extends the usual modal translations.) So, we can either think of a modal formalism, or of a piece of first-order logic, whichever seems more convenient. Let us call this language GSAL (Guarded Sequential Action Logic), viewed either way. It is weaker than A-GF in that we have fixed action predicates $R\mathbf{x},\mathbf{y}$: no permutation or duplication of arguments allowed. One deviant feature on the first-order side is its distinction between two sorts of predicate on tuples: state versus transition assertions. This distinction might seem empty in standard logic, but we'll give a principled account in terms of different semantic roles. One other point. We assume that GSAL has identity, but this is for convenience - and one can do without.

4.2 Bisimulation, Invariance and Safety

Bisimulations for GSAL are variations on 'potential isomorphisms' for the full first-order language. *Guarded bisimulations* are non-empty families E of finite partial isomorphisms between two models **M**, **N** with respect to the atomic state predicates, that are closed under domain restriction to sub-isomorphisms, and which satisfy zig-zag clauses along the atomic action predicates. E.g., using straightforward sequence notation to denote partial isomorphisms, one requires 'guarded choices' for any atomic action predicate R :

> *If* ***a**E**b*** *, and* $R^{\mathbf{M}}$ ***a'**, **c*** *with* ***a'*** *contained in* ***a**,*
> *then there exists* ***d*** *such that, for the* ***b'*** *contained in* ***b*** *which matches* ***a'**, both* $R^{\mathbf{N}}$ ***b'**, **d*** *and* ***c**E**d**.*
> *And vice versa with guarded choice from* ***N*** *to* ***M**.*

A first-order formula ϕ (**x**) is *invariant for guarded bisimulations* if, whenever **a** E **b** , then **M** |= ϕ (**a**) iff **N** |= ϕ (**b**). We call a first-order formula π (**x**, **y**) *safe for guarded bisimulations* if, whenever E is a guarded bisimulation (zigzagging for the basic action predicates of the language), the above zigzag clauses hold automatically for the new relation defined by π in the models **M**, **N**. Thus, safe formulas define transition relations that 'stay inside' our simulation semantics, i.e. our process realm. The basic property of GSAL is proved by a simultaneous induction.

Proposition *(1) All GSAL formulas are invariant for guarded bisimulations. (2) All GSAL programs are safe for guarded bisimulations.*

An adaptation of a known model-theoretic argument for modal logic shows a converse result as well.

Invariance Theorem *For all first-order formulas, the following two assertions are equivalent:*

> *(1)* ϕ *is invariant for guarded bisimulations*
> *(2)* ϕ *is definable in GSAL.*

A more laborious argument, again following a modal analogue, captures the safe operations. This amounts to expressive completeness for our key operations.

Safety Theorem *The safe operations are precisely those definable using (1) atomic action predicates, (2) tests for arbitrary state formulas, (3) projections, (4) relation composition, (5) union.*

We can vary a bit on this. Instead of all tests, atomic ones will do, if one adds an 'impossibility negation' ~ on actions. Safe programs describe unions (OR-trees) of finite sequences of multi-states linked by action steps or projections, with test assertions interspersed.

4.3 Basic Model Theory

Guarded bisimulation is like standard bisimulation, though technically a bit more difficult to visualize. Bisimulations now match finite sequences of states. There is a modified *unraveling* construction creating tree models – by marking of objects via paths <atom R**a**, **b**, selected object b_i , atom S**b'**, **c**, etcetera>. This can be used for various purposes, amongst others for interpolation and preservation properties. Here is a sample case used in the proof of the Safety Theorem. A formula ϕ(Q) is *totally distributive* in the displayed state predicate if its truth for the union of any family $\{Q_i \mid i \in I\}$ is equivalent to that for some Q_i separately.

Distribution Theorem *A GSAL formula is totally distributive in the state predicate Q**x** iff it can be defined in the form* $\langle\pi\rangle Q\mathbf{x}$*, where* π *is a safe program as described above whose test conditions on intermediate states do not involve the predicate Q.*

The dual nature of GSAL invites comparison with action predicates. A characterization for their total distributivity looks rather different. The tutorial will highlight such state-action differences occasionally.

4.4 Decidability and Axiomatization

GSAL is decidable, and it even has an effective Finite Model Property, because the action-guarded fragment has (via a direct Reduction Lemma for valid sequents).

Valid principles are much as in *-free PDL. For completeness several proofs exist (many-dimensional modal logic, other representation methods, proof-theoretic modification of GF decidability arguments).

4.5 Iteration and Fixed Points

Computation has a special interest in fixed points that can be reached in ω steps. In our first-order analysis, PDL-style operators suffice for all ω–fixed points $\mu Q \bullet \phi(Q)$ that can be computed with a matrix $\phi(Q)$ involving *one* suitable occurrence of the atom $Q\mathbf{x}$. Semantically, general ω-stability follows from Finite Distribution (i.e., ϕ holds of Q iff it holds of some finite subpredicate Q_0). The latter allows more general forms of definition with a finite number of suitable occurrences of Q. Full first-order logic has a simple syntactic normal form for finite-distributive operators:

$\mu Q \bullet \phi(Q)$ where the occurrences of Q-atoms in ϕ lie only in the scope of logical operators $\vee, \wedge, \exists$

For GSAL, a similar syntactic classification exists. It involves the existence of an AND-tree whose steps are safe actions, and whose nodes may now carry both Q-free test conditions and atomic tests involving Q.

Finite Distribution Theorem *For GSAL state-formulas ϕ, the following are equivalent: (1) ϕ is finitely distributive in Q, (2) ϕ states there exists one out of some set of finite action trees as just described.*

Finite distribution for action predicates is still open. Notice that defining state assertions by fixed points is not the same as defining new programs or actions by fixed points. (E.g., μ-calculus only has the former.)

We have a proof on probation to the effect that GSAL extended with fixed-point operators for state predicates defined by the above operations is decidable. (It generalizes the standard Fisher-Ladner filtration argument for PDL.) This is one instance of a general

Conjecture *GSAL with fixed-points is decidable.*

Indeed, a similar conjecture is around for the full GF. Finally, Lyndon-style preservation theorems for monotonic operations also generalize to GSAL, as do various Craig-style interpolation properties.

4.6 Moving to the full guarded fragment

The tutorial so far has developed the basic model theory of action-guarded first-order logic with the additional restrictions on guards imposed by GSAL. One can do the same analysis, first for the extension of GSAL which allows permutation and duplication of arguments in action predicates. Then, new 'safe' operations will appear, reflecting such permutations.

Next, one moves to the full Guarded Fragment, whose quantification pattern $\exists\mathbf{y}\ (G\ (\mathbf{x}, \mathbf{y})\ \&\ \phi\ (\mathbf{x}, \mathbf{y}))$ allows assertions linking up input and output states. In this case, partial isomorphisms will 'accumulate' in the zig-zag conditions for bisimulation. This will show in new 'safe' operations like **cum**(R) (x, yz) defined by 'Rx, y & z=x'. But the main structure of the preceding notions and results remains the same.

5 Parallel action, polyadic modality

5.1 Polyadic Modalities

The languages GSAL and GF were still sequential. To describe parallel action, one needs conjunctions of guard atoms, which are known to skirt the decidable-undecidable boundary. Using conjunctions, one can describe genuinely parallel actions, such as products:

RxS takes **ab** to **cd** when (1) **aRc** , (2) **bSd**

Many variations are possible here (including merges as in Process Algebra). To describe such compound transitions, GSAL must be extended at least to what may be called GPAL (Guarded Polyadic Action Logic) with *polyadic modalities* <R, S>. GSAL can express a local version viewing the two outcomes separately:

$$\langle R, S\rangle\,(A, B) = \exists \mathbf{zu}\,(R\mathbf{xz} \wedge S\mathbf{yu} \wedge A(\mathbf{z}) \wedge B(\mathbf{u}))$$

This reduces to a conjunction of GSAL formulas. But we want to combine compound action with assertions that describe the total result achieved, i.e. the stronger

$$\langle R, S\rangle\, Q = \exists \mathbf{zu}\,(R\mathbf{xz} \wedge S\mathbf{yu} \wedge Q\mathbf{zu})$$

This format is not guarded, or even pairwise guarded. (It is an interesting generalization of modal logic all the same.) General decidability results do not apply. We will present a simple example showing that

Proposition *Allowing arbitrary conjunctions of guards makes the guarded fragment GF undecidable.*

Thus, at least in principle, parallel action is connected with the decidability-undecidability frontier.

5.2 Complexity Thresholds

Closer analysis of the dangerous examples shows that their syntactic forms mix state predicates with action predicates. But this intuitive distinction seems equally justified for parallel action. Hence, we must backtrack from the current front-line in pushing decidability upward from GF. Instead, retreating to the action-guarded fragment A-GF, another way of striking out from there is to keep the separation into two predicate roles, but then, allow *arbitrary* guard conjunctions. (See Section 3 for sharpenedup syntax definitions.)

Conjecture *The action-guarded fragment with separate action and state predicates, but extended with arbitrary conjunctive guards is decidable.*

A natural proof strategy is the usual modal unraveling via finite tree models. In its wake, PGAL with polyadic modalities using any conjunctions of action guards would be decidable. Next, what happens if we add fixed point operators to this parallel action logic, on state or action predicates. Do we keep decidability? And, is there a difference between the two versions?

5.3 Parallel Bisimulation

Guarded bisimulations for GSAL can be extended to stricter bisimulations for the richer language PGAL. This requires additional zig-zags for joint actions. E.g.

If ***a*** *E* ***b*** *, and* *R**a'c'**, S* ***a''c''****, there must be* ***d', d''*** *with* *R**b'd'**, S* ***b''d''*** *such that* ***c'c''*** *E* ***d'd''***

This combines the results of two actions undertaken from a single collective state. We will discuss the fate of the earlier key results on invariance and safety in this setting. In particular, can we find expressively complete sets of operations for parallel actions?

Some Relevant References

[1] H. Andréka, J.van Benthem, & I. Németi, 1998, 'Modal Languages and Bounded Fragments of Predicate Logic', *J. of Phil. Logic* 27, 217 – 274.

[2] J. van Benthem, 1996, *Exploring Logical Dynamics*, CSLI Publications, Stanford.

[3] E. Grädel, 1997, 'On the Complexity of the Guarded Fragment', preprint, Dept. of Informatics and Mathematics, RWTH Aachen.

[4] Marco Hollenberg, 1998, *Logic and Bisimulation*, Ph.D. Thesis, Philosophical Institute, Utrecht.

[5] M. Marx & Y. Venema, 1996, *Multi-Dimensional Modal Logic*, Kluwer, Dordrecht.

[6] M. de Rijke, 1993, *Extending Modal Logics*, Ph.D. Thesis, ILLC, University of Amsterdam.

3 Guarded Quantifiers: Questions and Variations

The Guarded Fragment is a large modally inspired decidable part of first-order logic, whose 'instrument of variation' is bounding of the range of quantifiers by atoms. We consider some natural variations on the original Guarded Fragment, and present a number of new observations plus open questions.

3.1 Decidable fragments: extending GF

The Guarded Fragment of first-order logic allows only the bounded quantifier pattern

$$\exists \mathbf{y}\ (R(\mathbf{x}, \mathbf{y})\ \&\ \phi(\mathbf{x}, \mathbf{y}))$$

where the 'guard atom' G may have occurrences of the variables in the finite sequences **x**, **y** in any order and multiplicity. GF is decidable (Andréka, van Benthem & Németi 1998), indeed complete for doubly exponential time (Grädel 1997). This generalizes many standard modal logics. But, in order to translate, e.g., Since/Until temporal logic, or pair-arrow logic, one needs the larger Loosely-Guarded fragment (LGF) of first-order logic, which allows

> *conjunctions of atomic guards* in the above position R(**x**, **y**), provided each pair of distinct variables from **x**, **y** occur together in some guard.

(Pairs taken from the parameters **x** may have their co-occurrence outside of the scope of the existential quantification, as this may be imported up to equivalence.) LGF is decidable as well, by an extension of the original quasi-model argument for GF (van Benthem 1997A). Given its description, a better name for LGF might be the Pairwise Guarded Fragment. (Maarten Marx has suggested Packed Fragment as a better name.)

With pairwise guarded conjunctions, we seem to reach a clear complexity threshold. Not admissible, on pain of undecidability, are *arbitrary* conjunctions of guards:

Proposition GF extended with arbitrary conjunctions of guards is undecidable.

Proof (van Benthem 1997B) The 3-variable fragment of first-order logic is undecidable. Here is an effective reduction. Any 3-variable formula ϕ is satisfiable iff its guarded relativization $(\phi)^U$ to some new ternary predicate U is satisfiable in a *full Cartesian product* $U = D \times D \times D$. The latter can be expressed as the satisfiability of a formula

$(\phi)^U$ & CART(U)

where CART(U) $=_{def}$ (i) $\exists xyz\ Uxyz$ & (ii) $\forall xyz\ (Uxyz \rightarrow$ & U-followed-by "all permutations and identifications among {x, y, z}") & (iii) $\forall xyzuvw\ ((Uxyz\ \&\ Uuvw)$ $\rightarrow$ & U-followed-by "all selections of three variables from among {x, y, z, u, v, w}"). Note that the latter formula is in GF with added conjunctions of guards. ■

3.2 Decidable parallel action fragments: backtracking from GF

As we saw in Section 2, process logics may suggest other useful decidable fragments, which backtrack from GF, so to speak. Basic modal logic has a distinction between what may be called 'action predicates' Rxy that jump across accessibility links (from x to y), and 'state predicates' Px making some static assertion about the current state x. This distinction is obliterated in GF, whose predicates may be viewed indifferently as describing moves between states, or as descriptions of fixed states. Now, our idea is that by maintaining such a distinction, we can be more liberal with quantifier bounds – and in the limit, allow any conjunction at all. The motivation for making this extension in Section 2 was the study of parallel processes over tuples of local states. In this setting, we can interpret the negative result in Section 3.1 as saying that *unconstrained parallellism leads to undecidability*. But what if we design things more delicately?

Thus, we distinguish between state atoms $Q\mathbf{x}$ and action atoms $R\mathbf{x}, \mathbf{y}$ from the start. The comma in action atoms serves to separate *input states* on the left from *output states* on the right. The total language will have both 'action formulas' and 'state formulas', whose syntax can be manipulated independently. Here are some options.

GSAL1	Action formulas	$R\mathbf{x},\mathbf{y}$
	State formulas	$Q\mathbf{x}$, Booleans, $\exists\mathbf{y}\ (R\mathbf{x}, \mathbf{y}\ \&\ \phi(\mathbf{y}))$

This 'guarded state-action language' describes transitions from an old state to a new one, without cross-comparison between old states and new ones (as in a GF matrix $\phi(\mathbf{x}, \mathbf{y})$). The input-output distinction has various effects. E.g., action atoms Rx, y are very different from their converses Ry, x. Moreover, the above restriction to only action-guarded quantifiers has the effect of making every formula depend on some initial tuple of free variables. Thus, all formulas in GSAL1 are 'local': one cannot form closed sentences. As in ordinary modal logic, the natural definition of 'satisfiability' then refers to local truth at some tuple of states in a model. 'Global satisfiability', in the sense of truth at all tuples in a model, will turn out to be a much more powerful notion.

If some input states are to persist as output, we need further atoms like Rx, yx, while quantifiers $\exists\mathbf{y}$ only range over the *new* components of the output state. Naturally, a matrix statement may now refer to these new **y** plus the persistent **x**. Allowing all this turns **GSAL1** into **GSAL2**. Both languages are effective parts of GF, and thus inherit its decidability. Note that their syntax has no explicit operations on action predicates. Section 2 shows which *safe* operations can be added, however – mainly suitable 'choice' and 'composition' – without increasing the expressive resources of these fragments.

This is all 'sequential' action. Genuinely *parallel* versions enrich the action formulas by (unsafe!) *conjunctions*, while imposing various constraints on quantifier patterns. Quantifiers then collect all output states mentioned in conjunctions of atoms **&**R**x**, **y** . Moreover, to emphasize that the new objects form a coherent state, one may require the occurrnce of an atomic guard, either over the new **y**, or over the new **y** plus the persistent **x**. We list some options. But before proceeding, a warning may be in order. The purpose of all this variation is not to create a boring catalogue of formal languages – but rather, to demonstrate the effect of various expressive resources on decidability.

P-GSAL1	Action formulas	$R\mathbf{x},\mathbf{y}$, conjunctions
	State formulas	$Q\mathbf{x}$, Booleans, $\exists\mathbf{y}\ (\&R\mathbf{x},\mathbf{y}\ \&\ \phi(\mathbf{y}))$
P-GSAL1*	Action formulas	$R\mathbf{x},\mathbf{y}$, conjunctions
	State formulas	$Q\mathbf{x}$, Booleans, $\exists\mathbf{y}\ (\&R\mathbf{x},\mathbf{y}\ \&\ Q\mathbf{y}\ \&\ \phi(\mathbf{y}))$

As before, both languages allow only 'local' formulas, describing some tuple of states. The second fragment is obviously a part of the first. **P-GSAL2** and **P-GSAL2*** are defined analogously, but now allowing input states from **x** to reappear as output states. None of these languages lies inside GF (even though P-GSAL2* adds strong guards):

$\exists y_1y_2\ (Rx_1, y_1\ \&\ Rx_2, y_2\ \&\ Qy_1y_2)$ *is in P-GSAL1*, but not in GF*

$\exists y_1y_2\ (Rx_1, y_1\ \&\ Sx_2, x_2y_2\ \&\ Qx_2y_1y_2)$ *is in P-GSAL2*, but not in GF*

Now we make some observations about decidability.

<u>Proposition</u> Satisfiability in P-GSAL1* is decidable.

<u>Proof</u> We start from the original quasi-model decidability proof for GF (cf. Andréka, van Benthem & Németi 1998, van Benthem 1997A), with a universe of 'types' (sets taken from the finite family of relevant formulas) satisfying suitable closure conditions. From this, we constructed paths of types recording which formulas are true at any stage.

We modify this idea slightly, allowing types that describe desired behaviour on only some subset of the variables. Transitions extending a path are triggered explicitly by existential formulas $\exists \mathbf{y}\ (\&R\mathbf{x}, \mathbf{y}\ \&\ \phi(\mathbf{y}))$ occurring in the last type so far, with the $\mathbf{y}$ 'changing their values' – while the new end-type only has formulas *with free variables among the* $\mathbf{y}$. As a result, the 'life-time' of the input variables $\mathbf{x}$ ends at such a step. In the model construction, we use objects (π, x) as before, where x is among the active variables at the end of the path π . For the interpretation of predicates, we set

(a) a state atom $Q\mathbf{d}$ is only true of a tuple of objects if these lie on the same path, and were introduced *simultaneously* at the final transition, whose result-type contains the atom with the variables of the $\mathbf{d}$ (in the same order)

(b) an action atom $A\mathbf{d}, \mathbf{e}$ is only true if all its objects lie on the same path, and the atom with the corresponding variables plugged in (as in (a)) occurred in the conjunctive action prefix of some transition.

Each path has an associated assignment s_π defined on the variables in the last and one-but-last types of the path, sending x to the object (π^*, x) , where π^* is that subpath of π in which x was changed last. Clearly, action atoms will only hold between objects in the one-but-last and last stages. The Truth Lemma then says that

a (relevant) *state formula* ϕ holds under the assignment
of a path iff ϕ literally occurs in the last type of that path

As in the original decidability argument for GF, there are two cases of major interest. (**1**) First, consider state *atoms* $Q\mathbf{x}$. If $Q\mathbf{x}$ is in the last type of π , then – by our restriction on result-types of path transitions – its variables were among those affected by the final change. So, we have the above condition for truth of the atom. Conversely, if $Q\mathbf{x}$ is true under s_π , this can only have happened by a simultaneous introduction on π , with $Q\mathbf{x}$ explicitly present. (**2**) Now consider *existential quantifiers* $\exists \mathbf{y}\ (\&R\mathbf{x}, \mathbf{y}\ \&\ Q\mathbf{y}\ \&\ \phi(\mathbf{y}))$. If the latter occurs in the final type, then it is true – by an argument as for GF: one looks at the obvious path extension triggered by the existential formula. The crucial case is when such a formula is true under s_π : while it should occur in the last type of π . Let some tuple $\mathbf{d}$ of objects satisfy the specified action predicates, plus the state guard $Q\mathbf{y}$ and the matrix statement $\phi(\mathbf{y})$. By the definition of true action predicates, the $\mathbf{d}$ must have been introduced following the end of the current path. Moreover, as the state atom $Q\mathbf{y}$ holds, they were introduced together in one transition, resulting in one final type Δ (i.e., they do not lie on separate forks) containing $Q\mathbf{y}$. Call this extended path π^+ . Its s-assignment sends the variables $\mathbf{y}$ to the objects $\mathbf{d}$.

By the inductive hypothesis then, $\phi(\mathbf{y})$ occurs in Δ, the last type of π^+. But then, by an obvious existential closure condition on quasi-models, $\exists \mathbf{y}\ (\mathbf{\&}R\mathbf{x}, \mathbf{y}\ \&\ Q\mathbf{y}\ \&\ \phi(\mathbf{y}))$ occurred in the type before that, which was the final type of π. ■

We think that P-GSAL1 (without the guard condition on new state tuples) is decidable, too. But the above proof method does not work, since there is no guarantee that the new states introduced by a true existential quantifier $\exists \mathbf{y}\ (\mathbf{\&}R\mathbf{x}, \mathbf{y}\ \&$ form a 'simultaneous set' introduced in one parallel action step. (Different y might come from different steps.) On the other hand, various parts of the above argument seem to admit of generalisation. As for the two stronger languages P-GSAL2 and P-GSAL2*, we leave their decidability as an open question. Finally, note that the above proof is about local satisfiability only. It does not settle the decidability of *global* satisfiability. This issue will return below.

3.3 The danger zone: encoding tiling problems

Let us now approach these issues from a different angle, and see where undecidability strikes for sure. Consider the embedding of *tiling problems*. The undecidable task is to put coloured tiles on the infinite grid NxN, with some finite set of colours, and tiles having four coloured edges, subject to the constraint that adjacent tiles have the same colour along their boundary. First-order formulas expressing the relevant constraints have a definite P-GSAL flavour, with actions N (*go one step north*), E (*go one step east*) and state predicates Cx for the colours. Here are some examples. Adjacency of colours can be expressed by straightforward universal conditions of the form

$$\forall x: \quad \forall y\ (Nx, y \rightarrow (C_1x \rightarrow \vee C_2y))$$
$$\forall x: \quad \forall y\ (Ex, y \rightarrow (C_1x \rightarrow \vee C_2y))$$

where the unary predicates C_i describe the various possible kinds of tiles. General behaviour of colours is expressed by conditions of the form

$\forall x$: "at least and at most one C holds of x"

Next, the crucial *grid pattern* seen from x is expressed by the assertions

$$\forall x: \quad \exists y\ Nx, y \qquad\qquad \forall x: \quad \exists y\ Ex, y$$

and more importantly,

$$\forall x: \quad \forall yz\ ((Nx, y\ \&\ Ex, z) \rightarrow \exists u\ (Ey, u\ \&\ Nz, u))$$

These assertions lie in P-GSAL$_1$, modulo one unbounded universal quantifier in front. Let us call their conjunction TILE. Now it is not hard to prove the following

Fact NxN has a tiling iff TILE is satisfiable.

Proof Here is a sketch (for detailed arguments of this kind, cf. Spaan 1993, Blackburn, de Rijke & Venema 1998). Clearly, if a tiling exists, NxN *itself*, suitably expanded, verifies TILE. Conversely, consider any model for TILE. It is easy to define a map f from NxN, sending the origin to any point in the model, with the following property:

if y is a northern (eastern) neighbour of x, then N f(x), f(y) (E f(x), f(y))

To see this, use the last three formulas above repeatedly to construct a grid of squares x N y E u, x E z N u, which provides all necessary f-values. Then, a colouring for NxN meeting all constraints can be copied from the C-behaviour of the f-values. ■

3.4 Analysis: what causes undecidability?

What does this tell us? First, *the expressive power of parallellism comes close to encoding grids*, and hence undecidable tiling problems may arise. But the undecidable encoding does not quite lie in P-GSAL$_1$. We need *one unbounded universal quantifier in front* to make TILE work – whose dangers are well-known. Spaan 1993 shows how decidable modal logics can become undecidable with this simple addition. She states this in terms of adding a 'universal modality' to the logic, but also observes that one such modality in front, i.e., our earlier *global satisfiability*, would do the harm already. An alternative would use only those points (in models for TILE) reachable from some fixed origin by a finite number of E, N steps. This uses *transitive closure* of the relation N∪E, which is again outside our fragments – and even more dangerous for decidability. Spaan 1993 shows that the latter can embed the Σ^1_1-hard problem of 'recurrent tiling'. (For later use in Section 3.5, note that transitive closure is a fixed-point operator on relations, not on propositions.) Thus, a mixture of encoding grids plus some weak form of universal prefix quantification will make process logics undecidable.

Nevertheless, things are a bit delicate. For instance, adding one universal quantifier up front to the non-conjunctively-bounded *Guarded Fragment* does no harm! (Cf. van Benthem 1997B for similar observations for 'Sofia fragments' in extended modal logic.)

Fact Satisfiability in the GF with one universal prefix quantifier is decidable.

<u>Proof</u> Start with any type containing a few universally quantified guarded formulas x $\forall x\ \phi(x)$. Add all instances $[u/x]\phi$ (for the relevant variables u) to the types in the quasi-model. The original tree-model construction will still work as it stands – and it is easy to show that ϕ will hold for all tuples of 'path objects' of the form (π, u). ■

Recall that minimal modal logic plus a 'universal modality' remains decidable. Thus, it is the mixture of parallellism and universal quantification that generates undecidability. As to extensions of our observation about GF, Marx 1997 presents undecidable modal logics with characteristic *universal Horn* frame conditions. Therefore, allowing universal prefix quantification over larger tuples seems problematic already.

<u>Excursion</u> Maarten Marx has an interesting view of GF as a 'monadic language' defining properties of 'generalized objects', which may clarify the general situation discussed in this Section. The P-GSAL family generalizes the admissible 'properties' while trying to stay away from having genuine 'relations' between generalized objects.

Finally, there is another feature to our tiling argument. The formulas in TILE did not satisfy the syntactic constraint of the language *PGSAL1** , that new objects in quantification must come simultaneously *guarded* by some state predicate Q. This seems less serious. We can modify the definition of TILE by using a trivial unary predicate P at *all* points, as well as a trivial binary predicate Q at all point pairs:

$\forall x$: Px
$\forall xy$: Qxy

Without the (double) universal prefix quantifiers allowing this trivial obedience, it is unclear how to modify the necessary grid encoding, and get things right for proper tiling within the syntactic constraint on outputs imposed by PGSAL1*.

Clearly, adding parallel constructions (through conjunctive guards) comes close to undecidability. On the other hand, it need not do so in general (witness the decidability of PGSAL1), and it seems harmful mainly in league with universal prefix quantifiers. We leave the investigation of intermediate possibilities open. For the moment, we hope the preceding has sufficiently illustrated our main concern: probing the effects of expressive power on decidability in a sensitive manner, guided by guarded analysis.

<u>Remark</u> One can also investigate the above fragments for other nice logical properties. Here we just recall a point about *bisimulation*. As stated in Section 2, the distinction between state predicates and action predicates can be supported by assigning them two

different roles in the definition of guarded bisimulations. Action predicates regulate the picking of suitable tuples of objects in back-and-forth moves, while state predicates determine what counts as a 'partial isomorphism'. This has all kinds of effects on further model-theoretic properties. E.g., we have two kinds of monotonicity now.

Remark The notion of 'partial isomorphism' may have to change, too, because of the special status of *identity* in our fragments. Identity statements like $\exists y\ (Rx_1x_2, y\ \&\ \ldots\ \&\ y{=}x_1\ \&\ \ldots)$ circumvent the distinction between input and output states, and their effect is therefore hard to predict. But without identity, the characteristic bisimulation must be adjusted, even for the guarded fragment GF itself. The basic building blocks will now be binary relations between finite tuples of objects of the same length (which do not necessarily decompose into functions, or even binary relations as sets of ordered pairs) – or alternatively, binary relations between finite variable assignments.

3.5 Fixed-point extensions

Even strong guarded langauges like GF or LGF leave the decidability of several well-known decidable modal logics unaccounted for. The key example is modal K4. *Transitivity* of frames, expressed by the first-order formula

$$\forall xyz\ \ ((Rxy\ \&\ Ryz) \to Rxz))$$

is not pairwise guarded, as the variable combination xz is not guarded anywhere. Also, results like the decidability of the two-variable first-order fragment L_2 do not apply: transitivity needs essentially three variables. Then why is K4 (even easily) decidable? There are two possible lines of attack here. One extends the syntactic scope of GF and its ilk, to find still broader decidability results. We doubt this is feasible. Transitivity is dangerous: it is known to make first-order fragments undecidable (Börger, Grädel & Gurevich 1996). But there is a way-out, by an alternative diagnosis of K4's decidability, transcending first-order logic, while retaining the key role of bisimulation invariance. Recall that propositional dynamic logic (or the μ–calculus) is decidable. Now it is easy to see that K4 is precisely the logic of any iteration modality $[a^*]$, on which we do not impose any special frame restrictions at all. This is a genuinely different strategy. For, the PDL-language does not define transitivity! Like the basic modal language, it is *invariant for bisimulation* (the infinitary conjunctions needed to define iteration do not affect this), while transitivity is not. So we would need a counterpart to the μ–calculus.

Question Find decidable fixed-point extensions of the Guarded Fragment.

The current conjecture is that these exist, generalizing the modal μ–calculus, perhaps using the 'tree model property' highlighted by Moshe Vardi. But there is a subtlety here. The μ–calculus has only part of its possible fixed points, viz. those defined by recursion over *state predicates*! But one can also use fixed points for new program constructions, recursing over *action predicates*. E.g., transitive closure $\langle a^* \rangle p$ is mimicked by setting

$$\mu q \bullet \langle a \rangle p \ \vee \langle a \rangle q$$

But the natural recursion $a^* = a \cup a;a^*$ over binary relations is not expressed directly. I do not know if the μ–calculus remains decidable when adding the latter version. Likewise, state recursion and action recursion are two different ways of adding fixed points to GF and its ilk. For instance, the finite approximations for state predicate-based fixed point equations remain inside GF, whereas those for action predicates need not. To see the latter, note that substituting an arbitrary guarded formula for a guard atom need not produce a guarded formula (substitute $\neg$ Rxy for Axy in $\exists$y (Axy & Qy)). Only 'safe' formats $\exists \mathbf{y}\ (\alpha(\mathbf{x}, \mathbf{y})\ \&\ ...)$ have this substitution property, which unpack into iterated guarded quantifications. In this connection, recall the above discussion of tiling problems, where a transitive closure of action predicates N, E led to *undecidability*.

On a simpler note, for many practical purposes, it suffices to use *finitely distributive* operators, whose smallest fixed point occurs uniformly after at most ω iteration steps (cf. Sections 2 and 4, 7 below, which claim decidability for the state predicate case).

3.6 Finite models

Another topic of interest is the behaviour of GF and its variations on *finite models*. Andréka, van Benthem & Németi 1998 show that GF has many of the 'nice' properties of first-order logic. Typically, such properties are lost for full first-order logic in Finite Model Theory. But for GF, some of them transfer immediately to finite models, because of its *finite model property*. Indeed, for basic modal logic, we know all its nice meta-properties hold on finite models. Which general transfer principle is at work here?

3.7 Interpolation

Maarten Marx and Eva Hoogland have just shown that GF lacks Craig Interpolation. (It does have generalized interpolation in the sense of Barwise & van Benthem 1996.) This raises the issue what interpolation behaviour is exhibited by the above fragments – and whether such behaviour may serve as a guide toward identifying useful ones.

4 A Beth Theorem for Process Operations?

Here are some speculations about process operations by entirely classical means. The setting is not relational algebra (as in safety theorems), which stays inside single process graphs – but rather model constructions over process graphs.

1 Operations on process graphs

Think of processes as represented by process graphs (LTSs, polymodal Kripke models). Process operations are defined as operations F (A, B, ...) creating new graphs out of old ones, which must *respect bisimulation*. That is,

if A **bis** A', B **bis** B', ..., then F (A, B, ...) **bis** F (A', B', ...) .

Examples are *addition* A+B (joint rooting, offering the options of both), *sequential product* A•B (substitution at the end, continuing with B after A has been completed), or *parallel products* A**x**B (performing both processes simultaneously in the left and right components of ordered pairs). Further examples abound (polarity flip, merge, iteration).

One would like to find restricted *natural spaces of process operations*, preferably through some kind of semantic invariance, invoking a version of the Beth Definability Theorem. Thus, standard model theory would apply, as happens in Marco Hollenberg's 1998 Ph.D. thesis "Logic and Bisimulation" (Philosophical Institute, Universiteit Utrecht).

2 Defining operations by first-order theories

To represent matters in standard model theory, take models with new unary predicates A, B, ... for disjoint argument domains (with a union O (= 'old')) and a predicate N (= 'new') describing a disjoint value domain. Together, O and N exhaust the whole domain. The A, B, .. and N-components may be viewed as submodels for some language L describing the internal structure of the process graphs. In addition, to describe relevant relations between the argument and value domains, we add new predicates C 'connecting' objects in O to those in N . The latter may be identical with old objects (as happened in the above sum **+** and sequential product •), but they may also be new things, created by some construction (such as the ordered pairs in a parallel product **x**). Thus, we view the operation F as given by a class of models of this similarity type, where the additional vocabulary may satisfy a number of constraints. The above process operations may then all be specified in the following format:

I uniform first-order definition of the new objects in the value graph involving finite sets of objects in the arguments (possibly with some new object with a unique function, such as the new root added in A+B),

II uniform first-order definition of the L-predicates among the new objects in terms of the L-predicates among the old objects related to them in clause I .

Thus, F is defined by some first-order theory T_F whose models allow for this schema. Let us say, in this case, that T has **CDP**, the *constructive definition property*. When do first-order theories have this definitory character? We want a semantic criterion, matching some natural way of thinking about process operations.

3 Unique extension properties

Instead of bisimulation, let us first look at *isomorphism.* The semantic feature matching the above intuitive formulation seems to be this:

> Any partial L-isomorphism f matching the O-parts of two T-models has a *unique extension* to some total bijection f^+ between these models which is even an *isomorphism* with respect to the full language L+C.

Let us say, in this case, that T has **IEP**, the 'unique isomorphism extension property'. This says, more informally, that the semantic behaviour of the old objects 'enforces' that of the new objects introduced by the operation.

4 A Beth-type theorem?

Our natural conjecture would be the following kind of Beth theorem:

> *A first-order theory has CDP if and only if it has IEP.*

Although I got some way toward proving this, I did not yet arrive. We are trying to turn 'dependence' (in the sense of IEP) into explicit definability (in the sense of CDP). But what we have seems weaker than Beth's implicit definability. (For instance, even with a fixed O-part, the root in the construction can be chosen in different ways, and therefore, different isomorphic 'superstructures' are possible.) One technical trick uses (suitably) *saturated* models of T . The identity on the O-part must have a unique extension to the whole model. This implies that there cannot be non-trivial automorphisms of the value part extending the identity on the argument part. By familiar arguments, it then follows that objects must have unique definitions in the full language – allowing arbitrary sets of parameters in the O-part. This seems to tell us something about the above parts I and II.

But these definitions may still be 'local' in one single given model for T . Is then the additional force of the Unique Isomorphism Extension Property across *different* models that it enforces more uniformity on these definitions?

5 From isomorphism to bisimulation

I would also be happy with *strengthened* forms of IEP toward an equivalence with CDP. One natural strategy for this purpose would use *potential L-isomorphisms* instead of complete isomorphisms. Also, going back to the original motivation on process graphs, one would like to have good variants with *bisimulation* instead of (total or potential) isomorphism. Unfortunately, then, no unicity seems left (because of the much rougher identifications allowed in bisimulations) – as may be seen from inspection of the earlier examples of sum and products. What might still hold as a constraint is the existence of some unique *minimal extension* for the component bisimulations.

6 More general uses

This analysis might have several benefits. It would make the standard format for specifying process-algebraic operations more uniform from a model-theoretic viewpoint. Moreover, it would make the route taken in Marco Hollenberg's dissertation less ad-hoc. Its author assumes (with some pangs of conscience) that new process constructions involve states which are *finite sequences* of old objects, with some uniform finite bound on their length. It might be that this is an inevitable feature of any first-order treatment.

Addendum

This sketchy promissory note was written in early 1998. Sol Feferman has informed me in the meantime about two relevant earlier papers. (1) S. Feferman & R. Vaught, 1959, 'First-Order Properties of Products of Algebraic Systems', *Fundamenta Mathematicae* 47, 57–103. (2) S. Feferman, 1972, 'Infinitary Properties, Local Functors, and Systems of Ordinal Functions', in *Conference in Mathematical Logic, London '70*, Lecture Notes in Mathematics 255, Springer, Berlin, 63–97. In particular, (2) introduces an extension property for potential isomorphisms, while (1) studies when elementary equivalence for arguments implies elementary equivalence for values of model constructions. Algebraic products turn out to be a counter-example. By contrast, the usual operations in Process Algebra (including both its products!) all have this first-order preservation property. Thus, one may get a handle on different complexities for proposed process operations.

5 Modal Fixed Points and Bisimulation

These are some thoughts on Jon Barwise & Larry Moss' intriguing book *Vicious Circles* (CSLI Publications, Stanford, 1996) including a proposed simplification of their proofs, a mysterious analogy, and speculations about a broader moral. Larry Moss has some new
results that seem relevant to our discussion – but they have not yet been referenced here.

1 Characteristic Modal Formulas for Bisimulation Equivalence Classes

Barwise & Moss show that each modal model **M**, s has a *characteristic formula* $\phi^{\mathbf{M}}$ in an infinitary modal language with all set conjunctions and disjunctions. I.e., for all models **N**, t, we have an equivalence between the following two assertions:

(1) **N**, t $\models \phi^{\mathbf{M}}$
(ii) there exists a *bisimulation* between **M**, s and **N**, t (connecting s to t)

This is a variation on the well-known Scott Theorem for infinitary logic, with bisimulation taking the role of potential isomorphism, and the modal fragment that of the full first-order repertoire. (The same characterization was proved independently for countable models only in van Benthem & Bergstra 1995.) The method of proof goes as follows. Starting from atomic base descriptions, one works in ordinal rounds α . At each round, partial descriptions $\delta(\alpha, x)$ are generated for the worlds x in **M** . Let y range over all R–successors of x in **M** . Then the next description $\delta(\alpha+1, x)$ is defined to be

the conjunction of all statements $<> \delta(\alpha, y)$
together with the closure condition $[] \vee \delta(\alpha, y)$

At limit ordinals, one takes the obvious infinite conjunction of everything obtained so far. One can show that this construction will stabilize at some ordinal α^* (depending on the cardinality of **M**) after which no new descriptions are generated. The resulting formula $\delta(\alpha^*, s)$ is the characteristic formula, defining the bisimulation equivalence class.

2 Characteristic Modal Formulas via Fixed Points

What follows revolves around one simple observation. The above looks very much like the construction of a *fixed point*. Its template is a description E(**M**) for any modal model first given by Jankov and Fine in the 70s. [1] Here is the definition. Take a set of new proposition letters (different from those in the initial modal language) p_x : one for each world x in **M** . Moreover, for each x, let AT_x be the conjunction of all literals in the original language that hold at **M**, x . Here is the major tool in what follows:

> E(**M**) is the conjunction of all statements
> $p_x \rightarrow AT_x \ \& \ \mathrm{env}(\mathbf{M}, x)$, where
> env (**M**, x) is the formula $\mathop{\&}_{Rxy} \langle\rangle p_y \ \& \ [] \vee_{Rxy} p_y$

2.1 Computing uniform fixed points

Now, it is easy to establish the following description for all models of E(**M**) . Let the symbol ≡ indicate the existence of a bisimulation between two rooted models.

<u>Proposition 1</u>

For any two modal models **M**, s and **N**, t the following are equivalent:

(1) **M**, s ≡ **N**, t

(2) **N**, t can be expanded to a model for E(**M**) (i.e., the latter statement holds throughout **M**) such that the predicate p_s holds at world t .

Note that this amounts to the truth of some monadic second-order formula in **N**, t .

<u>Proof</u> <u>From (1) to (2)</u>. Define the predicates p_x in **N**, t by setting p_x (u) iff x≡u . All clauses of E(**M**) hold, by the back-and-forth conditions of modal bisimulation. <u>From (2) to (1)</u>. Define a relation E between worlds in **M** and **N** by setting x E u iff u satisfies p_x in the expanded model for E(**M**) . The latter's clauses ensure that u, x satisfy the same atoms, and that the back-and-forth conditions hold everywhere. ■

Next, there is a well-known intimate connection between truth of existential second-order formulas and the existence of *fixed points*. This suggest the following alternative version of the preceding result. Note that E(**M**), whose 'minimal reading' can be seen as an equivalence, may be viewed as a *simultaneous inductive definition* for the predicates p_x , all of whose clauses are syntactically *positive* in all p_y . Thus, the associated semantic approximation operator is monotone. Therefore, every modal model **N**, t has a greatest fixed point for the latter operator – say, GFP(**N**, E(**M**)) – whose obvious 'projections' to the predicates p_x satisfy E(**M**).

---

[1] These formulas were originally used to define axioms for special modal logics 'omitting' all p–morphic pre-images of some fixed finite set of frames. Van Benthem 1985 has further applications.

Proposition 2

For any two modal models **M**, s and **N**, t the following are equivalent:

(1) **M**, s $\equiv$ **N**, t

(2) **N**, t $\models$ (GFP(**N**, E(**M**)))$_{ps}$

From (2) to (1), this follows directly from Proposition 1. From (1) to (2), we must add the observation that, if any set of predicates satisfies the implications E(**M**) in **N**, with the root predicate p_s holding at t , then so does the greatest fixed point.

2.2 Special classes and specific definitions

The preceding observations have further implications. We can analyse modal fixed points in special cases of interest. In what follows, we use the basic modal language for ease of exposition. But everything we say transfers to a polymodal logic with many modalities. There are *two directions* here. Start with some class of models, and determine its modal invariants – or start with some class of modal formulas, and find the models which they can characterize up to bisimulation. We start with a characterization of the *finite models*.

Proposition 3

Each finite model is characterized by a formula of propositional dynamic logic.

Proof Consider any finite model **M** . Without loss of generality, we can pass to its contraction under the maximal bisimulation. The latter model has the following further property. Two worlds satisfy the same finitary modal formulas iff they are equal (otherwise, non-trivial bisimulations would occur after all). Therefore, by a standard combinatorial argument on finite models, each world x has a unique modal definition δ_x in **M**. But then, we can describe an explicit solution for the fixed point equations E(**M**), by setting the p_x equal to δ_x . More precisely, let $\mu_{\mathbf{M}}$ be the infinitary modal formula

$$[]^* E^\delta \& p_s$$

where $[]^*$ is the transitive reflexive closure of [] , and E^δ says that the δ_x satisfy E(**M**) . (Note that this formula is immediately definable in propositional dynamic logic.) It is evident from the above definition that **M**, s itself satisfies $[]^* E^\delta \& p_s$. Hence, any model bisimilar to it also does. Conversely, if an arbitrary model **N**, t satisfies $\mu_{\mathbf{M}}$, then the δ_x describe a set of predicates p_x as meant in the preceding propositions, which guarantee the existence of a bisimulation with **M**, s . (Incidentally, the prefix $[]^*$ only serves to guarantee that the δ_x solution works in the 'transitive closure' of the root t , but not necessarily throughout **N** – but that is enough for the argument.) ■

<u>Remark</u> One really needs an infinitary modal formula here. E.g., consider the single reflexive point. Its fixed point equation is merely p $\leftrightarrow$ <>p & []p . One can easily show the greatest fixed point for this, in any model, is the set of worlds satisfying the infinitary formula $[]^*$<>T . But the latter is not equivalent to a finite modal formula, as is easily shown by considering suitably large Kripke models of the form ({1, .., n}, 'successor'). Note that this solution is what automatically results from applying the above general solution schema $[]^* E^\delta$ to the model consisting of a single reflexive world.

It is also possible to derive a converse for the preceding proposition.

<u>Proposition 4</u>

Formulas from PDL characterize only finite models (up to bisimulation).

<u>Proof</u> Let ϕ characterize a model **M**, s . By the Finite Model Property for PDL, ϕ then holds in some finite **N**, t . So **N**, t $\equiv$ **M**, s, and ϕ characterizes a finite model. ■

<u>Theorem 5</u>

The finite models are precisely those that are characterized by PDL-sentences. [2]

This result can be improved to broader modal classes (e.g., ω–saturated ones). As an illustration, here is an instant proof of one of Baltag's theorems in Barwise & Moss. Here, the direction of interest reverses, going from some given class of modal formulas (viz. the finitary ones) to a corresponding model class.

<u>Proposition 6</u>

The models characterized by finite modal formulas are precisely
the finite well-founded ones.

<u>Proof</u> Any finite well-founded model satisfies some modal formula of the special form [] ...(k times)... [] $\perp$. Therefore, in the above formula $[]^* E^\delta$ & p_s , the initial modality $[]^*$ may be replaced by that $[]^k \perp$, and we have found a finitary characteristic formula. The converse is even faster. If a finitary modal formula ϕ characterizes some model, then it is satisfiable, and – by a standard modal unraveling argument – it must also be satisfied in some finite well-founded tree.

■

---

2 A speculation about the broader thrust of this result. Only one fixed point iteration is involved in the eventual characteristic formulas: namely, for the outermost reachability operator []* . This reflects the fact that fixed-point logics can replace nested iterations by one 'grand loop'. There should be a connection between the 'flatness' of the equations in E(**M**) and the well-known trick for coding subformulas by new propositon letters in the usual way, which only requires equivalences of forms p $\leftrightarrow$ <>q and p $\leftrightarrow$ []q .

Here are some further general issues.

Questions Which formulas characterize the well-founded models (the ZF sets)? [3]
When is the characteristic modal formula 'effective'?

There are also some further general questions, about the whole point in describing models or sets by modal formulas. Can we use known facts about modal logic to get interesting new lines on sets? For instance, do known properties of modal logic, such as *interpolation*, have some nice set-theoretic meaning? Or, what about the known PSPACE decision procedure for the minimal modal logic K? Or in line with our Sections 2, 4:

> Does the above analogy help us in matching *process operations* (combining modal models) with more standard *set operations*?

Van Benthem & Bergstra 1995 observe that the + of process algebra is just set-theoretic union (working on models). Can we compute the characteristic formula for a union *effectively* from those for the components? (This might be an interesting exercise in general fixed point logic.) What are the natural operations in this setting anyway?

3 Computing Fixed Points

3.1 Uniform fixed points

The above solution is not very pretty. Can we compute the greatest fixed point of E(**M**) in some nicer way? First, the above argument works (and hence characteristic formulas exist) because there is a *uniform ordinal bound* to the computation of a non-empty fixed point for E(**M**) *in any model* **N**, which only depends on the cardinality of **M** (not on that of **N**). This is worth noting, because not every fixed point equation has this uniformity property. [4] More precisely, the solution for E(**M**) in **M** itself will be found after at most $|\mathbf{M}|\bullet|\mathbf{M}|$ steps, as there must be a change in at least one unary predicate p_x at each approximation state. Hence, characteristic formulas for each predicate are found

---

[3] The 'De Jongh-Sambin Theorem' says that in the modal logic of all *transitive* well-founded models, every fixed point equation of the form $p \leftrightarrow E(p)$, where p occurs only 'boxed' in $E(p)$ (positive or not!) has an explicit solution. I think this is a reflection of the general Recursion Theorems in ZF, exploiting the well-foundedness of the models. Can we find similar general results in the present setting?

[4] E.g., the well-known smallest modal fixed point $\mu p\bullet[]p$ defines the well-founded part of the relation R in any model. Its computation length depends essentially on the latter's size.

at modal operator depth at most $|\mathbf{M}|\bullet|\mathbf{M}|$, i.e. $|\mathbf{M}|$ for infinite models (for finite ones: see below). Here is a formal statement, making the preceding analysis a bit more precise. Note, however, that the following argument makes no claim about a uniform bound for computing the *greatest fixed point* – only for *some* non-empty fixed point (with the appropriate proposition true at the root). So, there is an open question here. Also, the following proof is rather roundabout; and one would prefer a direct combinatorial one.

Fact 7

In any model $\mathbf{N}$, if a schema $E(\mathbf{M})$ reaches a non-empty fixed point at all, then it reaches one after at most $|\mathbf{M}|$ stages.

Proof Suppose some non-empty fixed point is reached in $\mathbf{N}, t$. Then by the earlier reasoning, there is a bisimulation between $\mathbf{M}, s$ and $\mathbf{N}, t$. Now $\mathbf{M}, s$ satisfies the explicit modal solution statement E^{δ} , where δ describes the satisfying modal formulas of depth $\leq |\mathbf{M}|$. But then, through the mentioned bisimulation, $\mathbf{N}, t$ must also satisfy the modal formula E^{δ} . And by its definition, that means that some non-empty fixed point was reached at stage $\leq |\mathbf{M}|$ inside $\mathbf{N}$. ■

For finite models, we might have a quadratic blow-up here: but we can do better. For, in the approximation sequence, whenever some predicate p_x does not change in some round, we can stick to its *previous* definition, instead of using the next layer. Thus, its complexity only increases when there is some real change. We have derived the general

Fact 8

Characteristic formulas for worlds (p_x) need only modal operator depth $|\mathbf{M}|$.

Specific examples may be computed by hand for simple cases, and then reveal further syntactic fine-structure – which we will forego here.

3.2 Fixed points at omega

To obtain a greatest fixed point for a formula $\phi(p)$, we compute a smallest fixed point for its dual $\neg \phi(\neg p)$ and then negate that. Often that greatest fixed point is found after ω rounds. Thus, for the single reflexive point, one works with the new equation

$$p \leftrightarrow \langle\rangle p \vee []p$$

which is modally equivalent to $\langle\rangle p \vee []\bot$. The latter has p only under existential modality and disjunction. Now all fixed point equations whose defining clauses for p have the latter only under <>, &, $\vee$ are *finitely distributive*, and hence they compute a smallest fixed point uniformly in at most ω rounds. For PDL-formulas, one can compute all relevant fixed points this way (cf. Kleene iteration). Thus, our analysis seems close to a proof that all characteristic formulas for finite models have this property.

Question When are characteristic formulas computable at omega?

Another interesting question would be to determine in some effective manner

Question Which general fixed point formulas have a uniform solution bound?
In particular, which formulas in the modal μ–calculus do?

4 Connections with Other Simulations

The above results give an *invariant* - in the standard mathematical sense - for models up to bisimulation. All models sharing the invariant $\phi_{\mathbf{M}}$ form a bisimulation equivalence class. But there are more results of this kind, that connect up with standard automata theory.

4.1 Automata and regular sets

Kleene's Theorem gives regular expressions $\kappa_{\mathbf{M}}$ as invariants for finite state machines **M** characterizing these up to *finite trace equivalence* (instead of bisimulation):

$\mathbf{N}, t \models \kappa_{\mathbf{M}}$ iff $\mathbf{N}, t$ has the same 'language yield' as $\mathbf{M}, s$.

This can also be stated in a modal language of 'finite-path formulas' (van Benthem & Bergstra 1995). In particular, we can compute $\kappa_{\mathbf{M}}$ with a *fixed point equation* as above, using predicates Y_x recursively describing the 'yield' of **M** starting from x . I did not find this particular point in the FSM chapter of Barwise & Moss, but I guess they do mean one can now generalize Kleene's Theorem to arbitrary machines (finite-state or not). Here is a further observation. The essential thing is that these new fixed-point equations are *simpler*. To describe the yield of **M** at x , one only needs

a disjunction of cases $\langle a \rangle Y_y$, with y running over the a-successors of x
(and this for all atomic actions a) .

That means that the fixed point will be reached *uniformly* in ω steps! This is true of course for all regular expressions (cf. the above point about propositional dynamic logic). Thus, the above characterization result about finite models may be interpreted as follows. It says that PDL does for them what regular expressions will do if you are interested only in their finite succesful sequences. I wonder what else one might get from this analogy.

Question Which topics in formal language theory match which issues in modal logic? [5]

---

[5] Also, what kind of set theory does one get if one makes this very rough identification?

4.2 Invariants and simulations

(1) Similar invariants exist for any simulation equivalence, e.g., generated graph isomorphism (van Benthem & Bergstra 1995). Is there a general connection between definitions of simulations and those for the characteristic formulas? Fixed-point analysis works for all simulations defined by pebble games (Barwise & van Benthem 1996, 'Interpolation, Preservation, & Pebble Games', to appear in *Journal of Symbolic Logic*).

(2) A very sweeping philosophical thought (though not for situation theorists). Perhaps, all language *is* just an invariant for analogies across situations? That is, we can reverse the usual order, and think of (infinitary) first-order logic as that language which all rational beings would inevitably *invent* which are born attuned to potential isomorphism.

(3) Kleene's result was striking because he invented a *finite notation* for his invariants. Modal invariants for finite models are also finite. What is going on really is that we introduce some effective notation for the relevant fixed points. For which models can we get invariants from some *effective fragment* of the modal language $ML_{\infty\omega}$? In particular, which models have their characteristic formulas inside the modal μ*–calculus*?

5 Connections with AFA Set Theory

Finally, there seems to be a very tight connection with the non-well-founded set theoy AFA. Is the above implicit in the relative consistency result for AFA vis-à-vis ZF?

5.1 Fixed points and the 'solution lemma'

The above fixed-point equations $E(\mathbf{M})$ are exactly the 'flat systems of equations' of the Solution Lemma, and so they drive the key AFA axiom. That one can make do with *flat* equations instead of iterated ones must have to do with an earlier standard trick. (In computing fixed points, we lose no generality by *coding* up all subformulas *by atoms*.) Also, the *uniform* bound on the fixed point seems related to the requirement that the solution must be a *set*. Any solution to 'unbounded' fixed point equations like the above $p \leftrightarrow []p$ (well-foundedness) would presumably give us a 'class'. Have we (including Barwise & MOss) then been doing essentially the same things twice?

5.2 Truth as simulation?

To conclude, here is a wild speculation. What is the essential role of modal formulas vis-à-vis models under bisimulation? A model is a possibly non-well-founded set. A formula is a well-founded object. Now infinitary modal formulas, as objects, can be *at least as* *complex* as the models they describe. (They may be even more complex, like when we use infinite conjunctions to describe finite models). This point has always bothered me. Finite models are characterized by their first-order theories, but the descriptive sentence is intuitively more complex than the model itself! So, what is the gain? One might just as well manipulate the model (rather than all subformulas, or other syntactic items). Perhaps the gain is in the *well-founded* structure of the formulas, which allows us to use some simple inductive techniques. (But is this really a line of defense available to Barwise & Moss, who advocate free-wheeling circularity all around?)

Whatever the answer, it seems to me that viewing models and formulas on a par has some advantages. We can think of a *truth definition itself* as a notion of simulation between models and formulas. Think of a language where all negations have been pushed inward. At the atomic level, 'truth' is simple embeddability (more or less half an atomic clause in potential isomorphism). Upward, quantifiers or modalities suggest natural zigzag conditions between the <>, [] successor structure in the syntax tree and R-successor structure in the model. [6] The result looks like the semantics behind Hans Kamp's Discourse Representation Theory, which has 'embeddability conditions' relating DRSs to actual models (cf. Kamp & van Eyck 1997). (It is also in the spirit of a Wittgensteinian picture theory of language.) Could one get further mileage out of this?

---

[6] *Bisimulation* itself, for instance, also works if one compares process models with *different* structure: provided some match makes sense between moves of the right sorts. This is the standard situation in practice. Much standard theory goes through then – including the Modal Invariance Theorem and the Modal Interpolation Theorem in Barwise & van Benthem 1996. Cf. Section 10.1 for further details.

6 Information Transfer across Chu Spaces

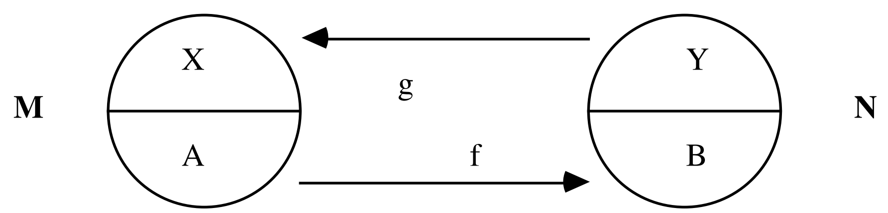
Chu spaces are a new model for information structure (cf. J. Barwise & J. Seligman, "Information Flow", Cambridge UP, 1997) and for mathematical structure in general (cf. Vaughan Pratt's ongoing work at the homepage http://boole.stanford.edu/live). Their properties are usually developed as a form of category theory. In this note, we show how they may also be viewed as models for a two-sorted first-order language, and we determine the exact flow of information across the natural Chu transforms. Our analysis is akin to that of process graphs via bisimulation and modal formulas.

1 Chu Spaces

A two-valued *Chu Space* is any structure (A, X, R) with two domains A, X and a binary relation R inside AxX. Examples: A = objects, X = sets, R = ∈ , or A = models, X = formulas, R = |= , or A = 'tokens', X = 'types', R = 'classification by'. Such spaces are naturally viewed as models for a *two-sorted first-order language* with variables a over (as we shall say) 'objects' and variables x over 'types'. Of course, one can also use other languages extending first-order logic here, such as infinitary or second-order ones. General Chu Spaces have a k-valued relation R (which makes them 'fuzzy' rather than crisp 2-valued classification structures), but in practice two-sorted examples predominate.

2 Chu Transforms

A *Chu transform* between two Chu spaces **M** = (A, X, ∈), **N** = (B, Y, ∈) (we shall use the same notation '∈ ' for convenience across Chu spaces) is a pair of functions f: A→B, g: Y→X (note the inversion in direction!) satisfying the following condition:



for all a∈A, y∈Y: *f(a)∈y* ⇔ *a∈g(y)*

There are motivations galore for this 'contravariant ' equivalence, for instance, in the logical theory of relative interpretation (cf. Barwise & Seligman 1997).

3 Preservation and Flow Formulas

What information is preserved in switching between Chu spaces connected by such a transform? We can view this as a standard question in model theory, asking for a *preservation theorem*. The following syntactic notion is obvious from some reflection on what we have, and do not have, in the above diagram:

> a *flow formula* is any first-order formula produced
> by the schema $a \in x \mid \neg\, a \in x \mid \& \mid \vee \mid \exists a \mid \forall x$

Flow formulas $\phi\ (a_1, .., a_k, x_1, .., x_m)$ can define many useful notions on Chu spaces: in general, relations between k objects and m types. Here are some examples:

$\forall x\ (\neg\, a_1 \in x \vee a_2 \in x)$	$\leq$	'object inclusion'
$\forall x\ (\neg\, a_1 \in x \vee \neg\, a_2 \in x)$	▭	'object incompatibility'
$\exists a\ (\, a \in x_1\ \&\ a \in x_2\,)$	***o***	'type overlap'

Let us call a first-order formula ϕ *Chu-preserved* if we have (with bold-face symbols indicating 'fitting' finite tuples of objects and types):

> **M, a, g(y)** $\models \phi$ *only if* **N, f(a), y** $\models \phi$
> whenever (f, g) is a Chu transform between **M** and **N**.

Of course, this notion also makes sense for non-first-order formulas ϕ.

<u>Proposition</u> All flow formulas are Chu preserved.

<u>Proof</u> This is a straightforward induction on the above definition, starting from the above characteristic Chu equivalence $f(a) \in y \Leftrightarrow a \in g(y)$ for literals. ■

<u>Application</u> Chu transforms are *monotone* with respect to object inclusion, as the latter relation was defined by a flow formula.

<u>Comment</u> We have described preservation in the 'f-direction' only. But in the opposite 'g-direction', we have the following syntactic description of preserved formulas (pushing negations inward for the equivalent implication "**N** $\models \neg\,\phi \Rightarrow$ **M** $\models \neg\,\phi$"):

> $a \in x \mid \neg\, a \in x \mid \& \mid \vee \mid \forall a \mid \exists x$

This outcome is precisely what one would predict by the obvious *duality* of Chu Spaces, where interchanging of the roles of A and X makes no difference.

4 Application: Rigid Chu Spaces

The preceding analysis systematizes several separate observations about Chu transforms. Take the following 'rigid Chu space' constructed by Pratt. Objects A = {1, 2, 3, 4}, types X = {x, y, z, u}, classification $\in$ as in the following diagram:

x: {1}	y: {2}
z: {1, 3}	u: {1, 2, 4}

Plotkin and Pratt have shown that the only Chu transform sending $(A, X, \in)$ to itself must be the *identity*. We can explain this by observing that each object $a \in A$ is *uniquely definable* by a flow formula, and therefore, it must 'land on itself' by our Proposition. Let the relations $\leq, =$ be as in Section 3 above. Here are the definitions:

(a) $\exists a_1 a_2 \,(a_1 \leq \mathbb{1} \,\&\, a_2 \leq \mathbb{1} \,\&\, a_1 = a_2)$ is unique for object $\mathbb{1}$

(b) $\forall x \,(\mathbb{1} \in x \vee \mathbb{2} \in x)$ is unique for object $\mathbb{2}$
(one uses the flow definition for object $\mathbb{1}$ here:
$\exists a \,(\mathrm{DEF}_{\mathbb{1}}(a) \,\&\, \forall x \,(a \in x \vee \mathbb{2} \in x))$.)

(c) $\forall x \,((\mathbb{1} \in x \,\&\, \mathbb{2} \in x) \rightarrow \neg\, \mathbb{3} \in x)$ is unique for object $\mathbb{3}$

(d) $\mathbb{4} \leq \mathbb{1} \,\&\, \mathbb{4} \leq \mathbb{1}$ is unique for object $\mathbb{4}$

5 Flow Preservation Implies Chu Tranform on Finite Models

Now let us convert the result, showing that the above is 'best possible'. Up to logical equivalence, *only* flow formulas are preserved under Chu transforms. We will formulate this as a preservation theorem in Section 6. But before proving this, we give a warm-up result inspired by an analogy with bisimulation and modal logic. (Some further aspects to this modal analogy that can be usefully exploited have been pointed out by Martin Otto.) The proof that follows here contains some key ideas for the later one.

Proposition For *finite* Chu spaces **M**, **N**, the following assertions are equivalent:
(i) there exists a Chu transform from **M** to **N**
(ii) every flow sentence true in **M** is also true in **N**

Proof (i) $\Rightarrow$ (ii) is a special case of our earlier Proposition. (ii) $\Rightarrow$ (i) works as follows. Enumerate A as $\{a_1, ..., a_k\}$ and Y as $\{y_1, ..., y_m\}$. We do one case of a stepwise construction for the desired function f. (The remaining case, as well as the construction of the contravariant companion function g, are similar.) The idea is that, progressively, f should assign some object $f(a) \in B$ to $a \in A$ which satisfies the same flow properties (where the latter may involve parameters for objects which have already been matched).

Suppose that no $b \in B$ satisfies all flow properties that hold for a_1 in **M**. That is, for some flow formula γ_b , we have that

$$\mathbf{M}, a_1 \models \gamma_b \quad \text{but } \textit{not} \ \mathbf{N}, b \models \gamma_b$$

Altogether, **M** then satisfies the flow formula (here we use $\exists \mid \&$ – closure)

$$\exists a \ \&_{b \in B} \ \gamma_b$$

But by condition (ii), the latter formula should hold in **N**. But, each $b \in B$ is disqualified as a witness for this, since it lacks 'its' conjunct γ_b .Therefore, by reductio ad absurdum, a 'good choice' b_1 must exist after all, and we can set

$$f(a_1) = b_1$$

This argument can be repeated to produce the successive values for f on all of A. Moreover, it can also be used in the opposite direction to find values for the function g, again maintaining the 'invariant' that flow formulas be preserved going from **M** to **N**. E.g., when searching for a matching type x for y_1, one assumes that each x_i fails for this purpose with a 'defect' δ_y , and then uses a flow formula of the form

$$\forall x \bigvee_{y \in Y} \delta_y$$

(with a dual use of $\forall \mid \vee$ – closure) to obtain a contradiction, going from **M** to **N**. ■

6 A First-Order Preservation Theorem

Instead of a standard preservation theorem, we formulate a slight strengthening in terms of 'generalized interpolation' (cf. Barwise & van Benthem 1998). Let us say that

> ϕ *implies* ψ *along Chu transforms* if always
> **M, a, g(y)** $\models \phi$ *only if* **N, f(a), y** $\models \psi$

Then we have the following result:

<u>Theorem</u> For all first-order formulas ϕ,ψ, the following statements are equivalent:

(i) ϕ implies ψ along Chu transforms

(ii) there exists a flow formula α such that $\phi \models \alpha \models \psi$

<u>Proof</u> (ii) $\Rightarrow$ (i) is again essentially the earlier Proposition. As for (i) $\Rightarrow$ (ii), assume that ϕ implies ψ along Chu transforms. First, define

$$FF(\phi) =_{def} \{ \alpha \text{ a flow formula} \mid \phi \models \alpha \}$$

It suffices to prove that

<u>Claim</u> $FF(\phi) \models \psi$

The required flow interpolant then exists by Compactness. So consider any countable model $\mathbf{N} = (B, Y, \in)$ for $FF(\phi)$. (This case suffices by the Löwenheim-Skolem theorem.) Let $Th_{\neg FF}(\mathbf{N})$ be the set of all $\mathbf{N}$-true *negations of* flow sentences. By a routine argument, using the closure of flow formulas under disjunctions, we have that

$Th_{\neg FF}(\mathbf{N}) \cup \{\phi\}$ is *finitely satisfiable*

Therefore, there is a (countable) model $\mathbf{M} = (A, X, \in)$ for $Th_{\neg FF}(\mathbf{N}) \cup \{\phi\}$, so that the following implication holds for all flow formulas γ:

$$\mathbf{M} \models \gamma \Rightarrow \mathbf{N} \models \gamma$$

Without loss of generality, we can even assume that $(\mathbf{M}, \mathbf{N})$ is a *recursively saturated model pair* with the same transfer property. But then we can mimick the earlier argument for finite models, this time, using the recursive saturation. Enumerate A as $\{a_1, a_2, ...\}$ and Y as $\{y_1, y_2, ...\}$. In the general case, suppose that some finite part of the pair (f, g) has already been constructed. Moreover, assume that all flow formulas whose free variables are set to objects a in the domain of f and types g(y) in the range of g in $\mathbf{M}$, and to the corresponding items f(a), y in $\mathbf{N}$, satisfy the following implication:

$$\mathbf{M, a, g(y)} \models \gamma \;\; \textit{only if} \;\; \mathbf{N, f(a), y} \models \gamma \qquad \#$$

Then we can extend this situation both ways. Here is the case for objects in $\mathbf{M}$ (that for types in $\mathbf{N}$ is similar). Let a^* be the first object in A without an f-value. Consider the recursive(!) set of all formulas of the following shape, where γ runs over flow formulas as in the preceding implication – except that there is one free object variable <u>a</u> on the right:

$$\gamma^M(\mathbf{a}, \mathbf{g(y)}, a^*) \rightarrow \gamma^N(\mathbf{f(a)}, \mathbf{y}, \underline{a})$$

This set is finitely satisfiable in the model pair $(\mathbf{M}, \mathbf{N})$, because for any finite number of flow properties γ_i of a^* in $\mathbf{M}$, we can form the flow statement (by $\exists \mid \&$ – closure)

$$\exists a \; \&_i \; \gamma_i(\mathbf{a}, \mathbf{g(y)}, a)$$

which is also true in **N** (by the earlier transfer implication #). So, we can find a value for <u>a</u> in B satisfying all these finitely many implications. But then, because of recursive saturation (our set is recursive, with only finitely many parameters from the domain), there is even some $b \in B$ satisfying this whole set of flow implications *simultaneously*, and we can choose this object to be the desired f-value for the object a^* . The argument in the opposite direction, producing a suitable g-value in **M** for the first virgin type $y \in Y$, is analogous, but now using the $\forall \mid \vee$ – closure of flow formulas.

Then, finally, we have $\mathbf{M} \models \phi$, **N** is a Chu transform of **M**, and so $\mathbf{M} \models \psi$: ■

<u>Remark</u> Sol Feferman (Stanford Logic Seminar, June 1998) has given an alternative proof for this preservation result using his interpolation theorem for many-sorted first-order logic. At its present state, this argument only covers Chu transforms with injective object maps. But it can presumably be modified to deal with the full case.

7 Variations and Extensions

The preceding result tells us precisely *how much* (or perhaps, *what little*) information is passed between Chu spaces that are related by their 'natural equivalence', at least for their *first-order* language. But we can vary the result to cover other cases of interest.

(1) The result goes through for special classes of Chu spaces, provided that these have first-order definitions. This holds in particular for *bi-extensional* Chu spaces, satisfying

$$\forall a \forall b\ (a=b \leftrightarrow \forall x(a \in x \leftrightarrow b \in x))$$
$$\forall x \forall y\ (x=y \leftrightarrow \forall a(a \in x \leftrightarrow a \in y))$$

In practice, this means that for bi-extensional Chu spaces, one can use two further atoms in flow formulas, without affecting preservation:

$$\neg\ x=y,\ a=b$$

(2) But also, the above proof itself can easily be modified to yield further preservations. For instance, if we know that the f-map in a Chu transform is *surjective*, then we can add *universal object quantifiers* $\forall a$ in the construction of flow formulas, and likewise, *existential type quantifiers* $\exists x$ if g is surjective.

(3) Finally, a first-order perspective also suggests *other equivalences* for Chu spaces, such as *elementary equivalence*, *potential isomorphism*, or pebble game variants thereof. From the perspective of information flow, there is no need for one model equivalence: the more structure preservation one can get, the better!

8 Infinitary Versions: Information Sequents

The preservation proposition of Section 3 still holds for arbitrary *infinitary* conjunctions and disjunctions in flow formulas. This explains the observations found in Barwise & Seligman 1996 on transfer (and non-transfer) of 'local logics' along 'infomorphisms'. In their terminology, let U, V be sets of types in **M** . We define true sequents:

$$U \mid\!\!-_{\mathbf{M}} V \qquad \text{if} \qquad \forall a\text{: } a \in \cap U \rightarrow a \in \cup V$$

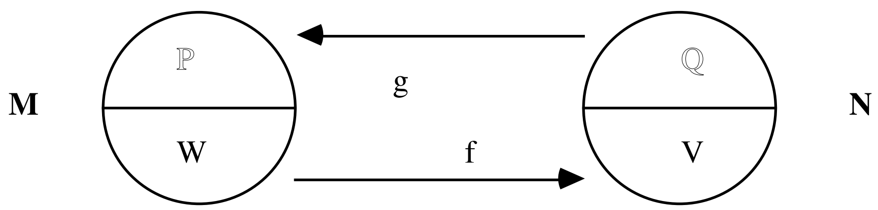
This infinitary definition is not an flow formula (as their maps f need not be surjective, universal object quantifiers are not allowed), and it is not preserved by Chu transforms. The implication will only hold in **N** on the image of f, the so-called 'normal tokens' in **N**. Thus, logically true sequents do not transfer in the f-direction. But they do transfer in the opposite g-direction, as the negation $\exists a\text{: } a \in \cap U \ \&\ \neg\, a \in \cup V$ *is* equivalent to an infinitary flow formula. (Barwise & Seligman do not consider further flow properties.)

This application increases the interest of an infinitary version of our preservation result. We conjecture that this is the case. But so far, we have only checked that the techniques of Barwise & van Benthem 1996 go through. These apply to model relations that can be cast in the form of pebble games. Applied to Chu transforms, this means the following. Instead of total maps, we now have a non-empty family of finite partial maps (f, g) between **M**, **N**, which satisfy the basic Chu equivalence for atoms, such that two back-and-forth properties hold, one extending each f-domain with an object from **M**, and one extending each g-domain with a type from **N**. Let us call these *potential Chu transforms*.

<u>Theorem</u> The above preservation theorem extends to formulas ϕ, ψ in the infinitary language $L_{\infty\omega}$, when we require preservation along potential Chu transforms.

9 Richer Chu Spaces: General Frames in Modal Logic

In modal logic, the natural Chu spaces are general frames $(W, \mathbb{P}, \in)$ with W a set of worlds, $\mathbb{P}$ a family of sets of worlds (the 'admissible propositions') and $\in$ membership. Here the natural equivalence is as in the following 'contravariant' picture (cf. Blackburn, de Rijke & Venema 1998, van Benthem 1985), where $f(w) \in)\ q$ *iff* $w \in g(q)$:



But here there is an additional requirement: f must be a *p-morphism* from **M** to **N**. I.e., it is a homomorphism for the accessibility relation $R^{\mathbf{M}}$, and it satisfies the zigzag clause

if f(w) $R^{\mathbf{N}}$ v, then there exists some $u \in W$ such that w $R^{\mathbf{M}}$ u & f(u) =v

Moreover, the map g is just the set-theoretic inverse f^{-1} on $\mathbb{Q}$ (landing inside $\mathbb{P}$!), which is a *homomorphism* with respect to the natural 'modal algebra' structure on $\mathbb{Q}$, $\mathbb{P}$. (This 'parasitic' nature of g is also known from Chu spaces in general.) Modal logicians have proved preservation theorems in this setting. But of course, more is preserved now, as the 'quality' of f, g is higher than in the above. In particular, flow formulas will now also allow the usual modal constructions, or more precisely:

atoms Rab | bounded universal object quantifiers $\forall$b (Rab $\rightarrow$

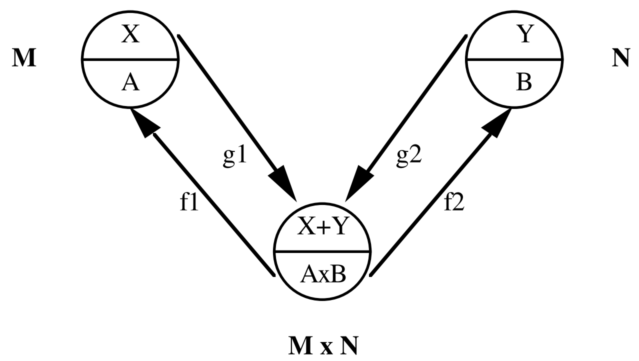
Combining this with the earlier syntax of flow formulas, we see that we get propositional literals p, $\neg$p, conjunction, disjunction, existential and universal modalities, plus arbitrary existential object quantifiers and universal propositional quantifiers. This includes all standard modal formulas, with a slight first-order and 'second-order' extra. This is surely an instance of a more general result, telling us how to 'load' our general Chu preservation with extra information from maps f, g that preserve special structure.

10 Constructions on Chu Spaces

The theory of Chu spaces gives a prominent place to (categorial) model *constructions*. One example is the *dual* operation taking (A, X, $\in$) to (X, A, $\ni$). Another is the product **M**x**N** used extensively in Barwise & Seligman 1996:

new object	AxB (Cartesian product)
new types	X + Y (disjoint union)
new epsilon	(a, b) $\in$ X^1 iff $a \in X$, (a, b) $\in$ Y^2 iff $b \in Y$

Here the picture of natural connections is as follows:



Here, preservation results might characterize formulas ϕ such that, if both $\mathbf{M} \models \phi$ and $\mathbf{N} \models \phi$, then $\mathbf{M}\mathrm{x}\mathbf{N} \models \phi$. (In particular, this holds for first-order Horn sentences.) But one does not want 'preservation' here so much as *combination* of information, or viewed in the other direction, *decomposition.* If we have a tight constructive definition of some operation on Chu spaces, then we can use it to reduce first-order evaluation.

Example 'Every Type is Inhabited'

Here is a simple semantic calculation from the given definitions:

$$
\begin{array}{ll}
\mathbf{M}\mathrm{x}\mathbf{N} \models \forall x\, \exists a\; a\in x & \text{iff} \\
\forall x\in X\, \exists a\in A\, \exists b\in B : (a, b)\in x \;\;\&\;\; \forall y\in Y\, \exists a\in A\, \exists b\in B : (a, b)\in y & \text{iff} \\
\forall x\in X\, \exists a\in A\; a\in x \;\;\&\;\; \forall y\in Y\, \exists b\in B : b\in y & \text{iff} \\
\mathbf{M} \models \forall x\, \exists a\; a\in x \;\;\&\;\; \mathbf{N} \models \forall x\, \exists a\; a\in x &
\end{array}
$$

So in this particular (Horn-definable!) case, the property does reduce to its presence in the components. In general, however, we don't expect this. Nevertheless, the example suggests an *effective component reduction* for arbitrary first-order statements $\phi^{\mathbf{M}\mathrm{x}\mathbf{N}}$:

(a) introduce a supply of marked variables with superscripts for A, B, X, Y

(b) replace object quantifiers $\exists a$ by $\exists a^A \exists b^B$, and replace corresponding atoms $a\in x$ in the formula by disjunctions $a^A\in x \vee b^B\in x$

(c) replace type quantifiers $\exists x$ by disjunctions $\exists x^X... \vee \exists x^Y...$

(d) replace (using the added markings) all 'heterogeneous' atoms $a^A\in x^Y$ or $b^B\in x^X$ by false

The result is a first-order formula which may be *separated* into an equivalent Boolean compound of separate fist-order assertions about **M** and **N**.

Sol Feferman has pointed out a more general background here. Chu dual and product satisfies the following preservation property (with '≡' for elementary equivalence):

if $\mathbf{M} \equiv \mathbf{M'}$ and $\mathbf{N} \equiv \mathbf{N'}$, then $\mathbf{M}\mathrm{x}\mathbf{N} \equiv \mathbf{M'}\mathrm{x}\mathbf{N'}$

Most operations in abstract process algebra have this feature (Hollenberg 1998). On the other hand, product spaces in the usual mathematical sense, whose objects are functions, do not (cf. the references given in Section 4). One obvious conjecture is that Chu *tensor product* as defined by Pratt in his model for linear logic, lacks this preservation property.

Preservation of elementary equivalence is a consequence of the above effective reduction. But having an effective decomposition seems stronger. So we want to know about both.

11 Co-limits and Generalized Evaluation

Dual and product were just two examples. The natural general construction is an *inverse limit* of families of Chu spaces which may have Chu transfoms running between them:

$$\begin{tikzcd} \mathbf{M}_i \arrow[r, "f"] & \mathbf{M}_j \arrow[d, "g"] & \mathbf{M}_k \arrow[l, "k"'] \\ & {} & \end{tikzcd}$$

'Objects' in the inverse limit $\mathbb{M}$ are tuples **a** having the right 'coherence': e.g.,

$$f\,((\mathbf{a})_i) = (\mathbf{a})_j, \text{ etcetera.}$$

This setting makes it much harder to do a 'logical decomposition' as above. It rather suggests that we generalize our perspective once more. One could think of evaluation of formulas in $\mathbb{M}$ as a *generalized semantic*s, where we have a family of models available instead of just one. We then sometimes shift (a bit like in some recent semantics for modal predicate logic) from looking for an object in one model to some image in another. This theme of 'long-distance evaluation' will return in Section 8 on 'information links', and, viewed as a strategy for 'decomplexifying' logics, in Section 10.3.

7 logical constants, semantic computation & simulation invariance

Johan van Benthem

ILLC, University of Amsterdam
October 1997

to appear in *Logica 98 Yearbook*
Institute of Philosophy, Czech Academy of Sciences.

Abstract

Semantic invariance approaches to 'logical constants' capture the important aspect of their 'topic-neutrality'. But these approaches tend to overgenerate, in that they admit all infinite Boolean combinations – which can hide a lot of unwarranted complexity. To advance further, we note that semantic invariances rather tell us something about the kind of *evaluation process* associated with logical constants. This process view leads us to impose a natural further constraint, of finite computability, which can be implemented over arbitrary models in a language-free manner. The result of such an analysis is a complete characterization of the logical constants that relate predicates and individual objects as precisely those definable in a standard first-order language. We also discuss ways of extending this analysis to more complex 'logical processes'.

1 Logical Constants, Semantic Invariances and Evaluation Processes

The logical expressions of a language are topic-neutral, and describe only abstract patterns in semantic models. Thus, they typically exhibit *invariance for permutations* of the universe of individuals (Tarski 1986, van Benthem 1986). But still very many expressions pass this test. More restrictive kinds of logicality arise by imposing invariance for less demanding semantic equivalence relations, sich as potential isomorphism, or bisimulation. Invariance may then be modified to a notion of 'safety' (preservation of back-and-forth behaviour), which allows for complete syntactic characterizations, e.g., of all safe first-order operations (van Benthem 1996, Ch. 5). Such results are attractive expressive completeness theorems, effectively enumerating all logical constants. A drawback common to all such invariance approaches, however, is their 'Boolean slippage': arbitrary infinite combinations of invarinat items satisfy the criterion. The reason is the symmetry of invariance, plus the usual inductive argument. (If one moves to different, asymmetric model relations to avoid this slippage, too many Booleans are lost, not just infinite ones.) Now, infinitary combinations are undesirable, as they encode a lot of unanalyzed structure that does not seem 'logical'. For instance, infinitary modal theories suffice for characterizing all sets (Barwise & Moss 1996)!

So, we have to find a further intuitive ingredient to logicality. Our analysis starts from the observation that semantic equivalence relations like the above may be viewed as 'simulations' between models, where the latter serve as process representations. Logical constants are naturally viewed as processes, viz. *evaluation procedures*. For instance, Tarski semantics defines the following evaluation process for first-order predicate logic. Its states are variable assignments, its basic moves are steps $=_x$ between assignments that agree up to their x–value (for the relevant variables x), while in between these, one can perform atomic tests on the current state (Groenendijk & Stokhof 1991). Now, our general suggestion is that all logical constants are evaluation procedures, and that 'logicality' also means computational 'simplicity' in some sense. In particular, this requires finite computation spaces. We will now implement this view more technically.

2 Semantic Computation: Approximating Models by Finite State Machines

Fix some finite predicate vocabulary, disregarding function symbols. Logical constants can be viewed as relations between these predicates, plus distinguished individuals. (Logical operations, like negation or composition of relations, may be subsumed here via their graphs – or via the use of models-cum-initial-assignments introduced below.) Thus, we identify a potential logical constant with some model class $\mathbf{C}$ in the relevant (predicate) similarity type. For instance, the clas of models $\mathbf{M} = (D, P, Q)$ where P, Q are unary predicates with a non-empty intersection encodes the logical notion of 'overlap'. More generally, we also consider pairs $(\mathbf{M}, s)$ with s a variable assignment. For instance, the class of pairs $(\mathbf{M}, s)$ with (D, P, Q) as above, and s an assignment sending one single variable x to some object in both P and Q , naturally encodes the logical operation of intersection. Next, we seek a link with semantic computation. Fix some finite number of variables k . After all, any computation process uses only a fixed number of registers for accessing objects in the domain of the relevant models.

2.1 Evaluation states: assignments modulo zigzag equivalences

It seems rasonable to identify our computational states with 'current workspaces', being k-assignments from these variables to objects in our model. But this may still be an infinite set (viz. if the domain of individuals is infinite), and not all differences between k–assignments need be relevant for our intended computations. Given any model $\mathbf{M}$, we therefore define a family of equivalence relations between states $\sim_d$ by induction:

$s \sim_0 t$	iff	the relation $s \circ t^{-1}$ is a partial isomorphism (i.e., s, t satisfy the same atomic facts place-by-place)
$s \sim_{d+1} t$	iff	$s \sim_d t$ and for each variable x and object d in $\lvert\mathbf{M}\rvert$, there exists an object e in $\lvert\mathbf{M}\rvert$ with $s[x:=d] \sim_d t[x:=e]$, and vice versa from right to left.

Note that these are language-free relations (introduced e.g. in Chang & Keisler 1970). We can consider their equivalence classes as appropriate abstract computation states. In particular, we do not need the concrete k–assignments displaying domain objects when computing Tarski's truth definition, since evaluation need not touch the actual objects (provided that we have access to the outcome of all relevant atomic tests). Of course, the larger the index d , the more information we get from $\sim_d$ about our current model.

2.2 Linguistic analysis: types up to some quantifier depth

There is a well-known 'linguistic' definition for the preceding relations.

Proposition 1 The following assertions are equivalent for all models and assignments:

(i) $s \sim_d t$

(ii) $\mathbf{M}, s \models \phi$ iff $\mathbf{M}, t \models \phi$ for all first-order formulas ϕ in k variables up to quantifier depth d

Proof (i) $\Rightarrow$ (ii) requires a well-known induction on the quantifier depth d of ϕ . For (ii) $\Rightarrow$ (i), we also use the logical finiteness of the latter first-order language. ■

<u>Corollary 2</u> Let $\mathbf{x}$, $\mathbf{y}$ be sequences of objects, both of length k . There are obvious corresponding k–assignments $s_{\mathbf{x}}$, $s_{\mathbf{y}}$. The relation $E^d(\mathbf{x}, \mathbf{y})$ defined by $s_{\mathbf{x}} \sim_d s_{\mathbf{y}}$ is first-order definable ('having the same d–type in k variables').

2.3 Model approximations by filtration

Next, we define a family of model approximations $\mathbf{M}^k{}_d$, which are finite Kripke models for modal logic, or more computationally (' annotated') finite state machines:

states	all $\sim_d$ equivalence classes s^d
transition relations	$s^d =_x t^d$ iff $s' =_x t'$ in $\mathbf{M}$ for some $s' \sim_d s$, $t' \sim_d t$
atomic valuation	$s^d \models Px_1...x_k$ iff $\mathbf{M}, s \models Px_1...x_k$

The valuation is well-defined, since any two n–equivalent states agree on all atoms. Note also that $\mathbf{M}^k{}_d$ is not an ordinary first-order model. There are no objects, and atoms are directly interpreted by their truth values at states without looking up tuples of objects in the usual Tarskian manner. This is precisely what we have in Kripke models for *modal formulas*. More precisely, the $\mathbf{M}^k{}_d$ are multi-S5 models. Again, one can look at this construction purely structurally. The reader will find it helpful to draw a concrete sequence of n–approximations, seeing how these reflect the structure of given first-order models. (A good example is (IN, <) with k=3 .) Incidentally, this is a new source of concrete models for modal logic, quite different from the usual examples. Here is an observation that we shall need later on:

<u>Proposition 3</u> Each individual model $\mathbf{M}^k{}_d$ is finite. Moreover, there are only finitely many different models $\mathbf{M}^k{}_d$ up to isomorphism. Both these finite numbers have upper bounds which are effectively computable from k, d .

<u>Proof</u> This is a simple calculation from the given definitions. It reflects (in non-linguistic terms) the logical finiteness of the above predicate language. ■

Thus, over the universe of all models, there are only finitely many 'projections' $\mathbf{M}^k{}_d$. Let us call this finite set $\mathbb{M}^k{}_d$. It is easy to show that not all multi-S5 Kripke models up to this size are filtrations of first-order models.

<u>Question</u> Is there a good *representation theorem* singling out those who are?

Even with such a result, it would still be *undecidable* if a modal model is representable in this way. Otherwise, one could decide universal validity for any first-order formula ϕ by surveying all appropriate finite modal candidate models for it, up to the above-mentioned effectively bounded size for its number of variables and quantifier depth.

2.4 Linguistic analysis: relating truth across filtrations

On the above models, first-order formulas behave just as modal formulas, with existential quantifiers $\exists x$ as existential modalities $\langle x \rangle$ for each of the k variables. One precise connection is a well-known *Filtration Lemma* from modal logic.

Proposition 4 For all first-order formulas ϕ with k variables and quantifier depth at most d , $\mathbf{M}, s \models \phi$ iff $\mathbf{M}^k{}_d, s^d \models \phi$

Proof Induction on the depth of ϕ . The atomic step is by the definition of a valuation. Boolean cases are routine. Next, consider the existential quantifiers. If $\mathbf{M}, s \models \exists x\, \psi$, then there exists some assignment $t =_x s$ with $\mathbf{M}, t \models \psi$. By the inductive hypothesis, $\mathbf{M}^k{}_d, t^d \models \psi$. By the above definition, $t^d =_x s^d$, whence $\mathbf{M}^k{}_d, s^d \models \exists x\, \psi$. Conversely, suppose that $\mathbf{M}^k{}_d, s^d \models \exists x\, \psi$. By the truth definition for the existential modality $\langle x \rangle$, there is a state t^d with $s^d =_x t^d$ and $\mathbf{M}^k{}_d, t^d \models \psi$. By the definition of $=_x$ on equivalence classes, there are states $s' \sim_d s$, $t' \sim_d t$ with $s' =_x t'$ in $\mathbf{M}$. Now, by the inductive hypothesis, $\mathbf{M}, t' \models \psi$. Then also $\mathbf{M}, t \models \psi$, by Proposition 1. But then, by the standard first-order truth definition, $\mathbf{M}, s' \models \exists x\, \psi$, and once more by Proposition 1, we have the desired outcome that $\mathbf{M}, s \models \exists x\, \psi$. ■

Proposition 4 only tells us how to relate truth of formulas up to quantifier depth d . But there is a more general result allowing us to reduce evaluation of arbitrary first-order formulas in filtration models $\mathbf{M}^k{}_d$ to what happened in the parent model $\mathbf{M}$. One can *translate backwards* from $\mathbf{M}^k{}_d$ to $\mathbf{M}$, by faithfully transcribing the above definition of states and accessibilities For this purpose, we define (cf. Corollary 2 for notation):

$$
\begin{array}{lcll}
(\phi)^{\#} & = & \phi & \text{for all atoms } \phi \\
(\neg\,\phi)^{\#} & = & \neg\,(\phi)^{\#} & \\
(\phi\&\psi)^{\#} & = & (\phi)^{\#}\ \&\ (\psi)^{\#} & \\
(\exists x_i\,\phi)^{\#} & = & \exists x_1' \ldots\ \exists x_k'\ (E^d(x_1, .., x_k, x_1', ..., x_k') & \\
 & & \&\ \exists x_i\,(\phi)^{\#}(x_1', .., x_i, .., x_k'))\,. &
\end{array}
$$

Note that the latter formula is indeed first-order, using the finiteness of k, n–types. Now, a straightforward induction establishes the following

Proposition 5 For arbitrary modal formulas ϕ , $\mathbf{M}^k{}_d, s^d \models \phi$ iff $\mathbf{M}, s \models (\phi)^{\#}$

We can check that this generalizes the Filtration Proposition 4 by observing that, for all formulas ϕ up to quantifier depth d, the equivalence $\phi \leftrightarrow (\phi)^{\#}$ is universally valid.

We conclude by noting that the same constructions and arguments work on any model **M** with some distinguished assignment s – the basic setting for Tarski semantics. The latter lands in $\mathbf{M}^k{}_d$ as s^d, which we can think of as a distinguished 'starting state'.

3 Logicality as Bisimulation Invariance in a Finite Computation Space

In this technical setting, we can sharpen up our general analysis. A 'logical' relation is a semantic computation process. This means two things. (1) On any model, it only uses a *fixed finite workspace*, no matter how large that model is. (2) It does not distinguish models with 'the same' associated process: i.e., whose associated workspaces are related by a standard process equivalence. For the latter purpose, we use an obvious candidate.

3.1 Basics of bisimulation and modality

There is a strong case for *bisimulation*, defined as usual (cf. van Benthem 1996), as a basic equivalence preserving both external output and internal choices of a process, across many fields (logic, computer science, game theory). We know, in particular, that modal formulas are invariant for bisimulation. Of various converse results, we mention

Lemma 6 Finite models are modally equivalent iff they are bisimilar.

Proof Cf. any modern textbook. From right to left, this is a straightforward induction on modal formulas. From left to right, one can take modal equivalence between states as the bisimulation. The back-and forth clauses use the finiteness essentially. ■

Also useful is the following simple consequence.

Lemma 7 Let A be some finite set of finite modal models. Let B be any bisimulation-closed subset of A . Then B has a modal definition in A .

Proof Consider any model **M** in B , and any model **N** in A–B . The two are not bisimilar, because of the closure condition on B . By Lemma 6, there is then some modal formula $\mu_{\mathbf{M},\,\mathbf{N}}$ true in **M** and false in **N** . The conjunction of all these formulas with **N** running over the finite set A–B is a modal formula true in **M** but false throughout A–B . Then the disjunction of the latter formulas, with **M** running over the finite set B , is the required modal definition for B in A . ■

3.2 Defining logicality as finite process invariance

The above two requirements on logicality now naturally combine into one. Consider any class **C** of models with distinguished assignments, standing for a putative logical relation. As earlier, we assume that models come with some distinguished assignment.

We call any such model class *finite-bisimulation-invariant* (FBI) if there exist two natural numbers k, d for which the following invariance condition is satisfied:

> *for all models* $\mathbf{M}\in\mathbf{C}$*, and for all models* $\mathbf{N}$ *,*
> *if* $\mathbf{M}^k{}_d$ *is bisimilar with* $\mathbf{N}^k{}_d$ *, then* $\mathbf{N}\in\mathbf{C}$

In this formulation of the FBI property, the bisimulations between $\mathbf{M}^k{}_d$ and $\mathbf{N}^k{}_d$ are always taken to connect the two distinguished starting states s^d, t^d – even if the latter have not been mentioned explicitly.

4 From Logicality to Definability

4.1 First-order definability

Here is our main result, which amounts to the following syntactic characterization. (The term 'first-order definable' refers to definability by one single formula.)

Theorem 8 A class of models is FBI iff it is first-order definable.

Proof First-order definable classes of models are FBI. Suppose that ϕ defines $\mathbf{C}$. Let k indicate all the variables occurring in ϕ , and let d be the quantifier depth of ϕ . Suppose that $\mathbf{M}, s \in \mathbf{C}$ satisfies ϕ . By Proposition 4, $\mathbf{M}^k{}_d , s^d \models \phi$ as well. Now let $\mathbf{N}$, t be any model such that $\mathbf{M}^k{}_d, s^d$ is bisimilar with $\mathbf{N}^k{}_d, t^d$. By the invariance of modal formulas under bisimulations, we get $\mathbf{N}^k{}_d , t^d \models \phi$. Once more by Proposition 4, $\mathbf{N}$, t satisfies ϕ as well, and hence – since ϕ defined $\mathbf{C}$ – $\mathbf{N}, t \in \mathbf{C}$.

Conversely, consider the d,k–projections of all models in our FBI class $\mathbf{C}$. This is a finite subset $\mathbf{C}^k{}_d$ of the finite class of all finite models $\mathbb{M}^k{}_n$. By Lemma 7, the finite bisimulation closure of this set has a modal definition μ . (Note that all models in the latter closure are bisimilar to some member of $\mathbf{C}^k{}_d$.) This modal formula *as it stands* need not be the required first-order definition. (Proposition 4 only applies to formulas up to modal depth d , and we have no reason to think μ is of the latter kind.) But by Proposition 5, we can *translate backwards* from $\mathbf{M}^k{}_d$ to $\mathbf{M}$, and use the first-order formula $(\mu)^\#$. The latter indeed defines our class $\mathbf{C}$. First, if $\mathbf{M}, s \in \mathbf{C}$, then $\mathbf{M}^k{}_d, s^d$ is in $\mathbf{C}^k{}_d$, and hence it satisfies the modal formula μ . By Proposition 4 then, $(\mu)^\#$ must hold in $\mathbf{M}$, s . Conversely, assume that $\mathbf{M}$, s $\models (\mu)^\#$. By Proposition 5, its approximation $\mathbf{M}^k{}_d, s^d$ satisfies μ . Then, by the above construction of the modal formula μ , this means that $\mathbf{M}^k{}_d, s^d$ is bisimilar to $\mathbf{N}^k{}_d, t^d$ for some model $\mathbf{N}$, t in the class $\mathbf{C}$. But then, by the definition of the FBI property, $\mathbf{M}, s \in \mathbf{C}$ as well. ■

4.2 Relaxing the bounds on computation

This is not the only result that can be extracted from this style of analysis. In particular, our restriction to some fixed finite bound on the computation space rules out cases with genuine iteration, such as fixed-point operators. For instance, computing the operation of *transitive closure* **tc**(R) (x, y) involves computing through finite spaces whose size may depend on the arguments x, y. This case may be covered, however, by the following relaxation of the above FBI property, shifting its quantifiers somewhat:

$$\exists k \ \ \forall \boldsymbol{M}, s \in \boldsymbol{C}\ \exists d \ \ \forall \boldsymbol{N}, t:$$
$$\text{if } \boldsymbol{M}^k{}_d, s^d \text{ is bisimilar to } \boldsymbol{N}^k{}_d, t^d, \text{ then } \boldsymbol{N}, t \in \boldsymbol{C}$$

An easy modification of the preceding proof in Section 4.1 shows that this weaker property holds for a class of models **C** if and only if the latter is definable by a *countable disjunction of* first-order formulas. As countable disjunctions may be highly non-effective, however, we feel this outcome still cannot be the final word.

4.3 Other states over first-order models

The preceding analysis of transitive closure is still unsatisfactory, as it does not capture the *uniform finiteness* of the process involved. This latter is the computation of a fixed-point with a fixed scheme whose approximation sequence 'stabilizes' after ω rounds. One way of representing these takes richer states (s^d, i) combining the above s^d standing for an 'environment' that yields replies to tests, with the 'current instruction' i . Such states occur in computations by Turing or Register machines. Here is a program checking whether the transitive closure of the binary relation R connects x with y :

1: IF **Rxy** THEN 2 ELSE 4   2: SUCCESS   3: FAILURE
4: IF **∃z Rxz** THEN 4 ELSE 5   5: SET **x:= εz• Rxz** ; GOTO 1

This program terminates succesfully just in case tc(R)(x, y) . It may diverge or fail otherwise, depending on the model. These actions can be described in terms of (s^d, i) state models with d=1 (no test for the program reaches greater depth), while arrows between the i's encode possible further activity. We then need a notion of bisimulation on such product models, which we will not pursue here. Note also that we need a new indeterministic atomic action x:= εz• Rxz ('x becomes some successor of its old self'), different from the random changes in x–values that sufficed so far. Alternatively, we can make use of the fact that fixed-point computations correspond to expandability of the original model with certain *additional predicates*, and complicate our notion of state s^d accordingly. We leave the analysis of fixed-point computations to another occasion.

5 Points for Discussion

5.1 This analysis is close to the usual characterization of first-order logic in terms of Ehrenfeucht games. We have merely 'rearranged the pieces' to throw some new light.

5.2 There is also a close connection to algebraic-style generalized **CRS**–models for first-order logic, and their representation theory.

5.3 Can one give a similar analysis for definability in first-order logic plus monotone (or just ω–continuous) fixed point operators? This would be an interesting step toward a purely *semantic* analysis of the notion of computational 'algorithm'.

5.4 How does our analysis of logical relations between individuals and predicates extend to relations at higher type levels?

5.5 Are there good *representation theorems* for finite modal models as $\mathbf{M}^k{}_d$'s?

5.6 Develop some standard model theory of d,k–approximations. Can each $\mathbf{M}$ be retrieved as an inverse limit of its approximations $\mathbf{M}^k{}_d$, plus their natural connections?

5.7 The only atomic actions allowed in our analysis of semantic computation are random shifts in single registers (cf. the relations $\sim_x$), and tests for atomic formulas. One might consider richer repertoires, such as multiple assignment, and choice of new values constrained by some atom (e.g., 'let x become one of its own R-successors'). What will happen to our previous analysis? What happens if we throw in further infinitary regular constructions, like iteration?

5.8 Is there a link with the category-theoretic analysis in Butz & Moerdijk 1997?

6 References

J. Barwise & L. Moss, 1996, *Vicious Circles*, CSLI Publications, Stanford.
J. van Benthem, 1986, *Essays in Logical Semantics*, Reidel, Dordrecht.
J. van Benthem, 1996, *Exploring Logical Dynamics*, CSLI Publications, Stanford.
C. Butz & I. Moerdijk, 1997, 'A Definability Theorem for First-Order Logic', to appear in the *Journal of Symbolic Logic*.
C.C. Chang & H.J. Keisler, 1970, *Model Theory*, North-Holland, Amsterdam.
H.C. Doets, 1996, *Basic Model Theory*, CSLI Publications, Stanford.
J. Groenendijk & M. Stokhof, 1991, 'Dynamic Predicate Logic', Ling. & Phil. 14, 39-100.
A. Tarski, 1986, 'What Are Logical Notions?', in J. Corcoran, ed., *History and Philosophy of Logic* 7, 143-154.

8 Information Links and Logical Transfer

Information can reside in a number of different but connected situations.
We discuss the logical structure of information flow across these links,
using 'generalized consequence relations' in a modal logic framework.

1 Information Networks

One situation can carry information about another, provided there is sufficient 'connection' between the two. This idea is the core of Fred Dretske's analysis of information flow, as developed further in a logical vein by Barwise & Seligman 1996, Israel & Perry 1991. Such connections can be 'extrinsic' (due to regularities that happen to hold in this world), but also 'intrinsic': based on structural similarities between the situations. One can model both by *information network* plus useful *links* between them. (Another source for this idea is Michiel van Lambalgen's work on information flow across various approximations of visual scenes.) Unlike Barwise & Seligman, we do not assume that these links are of one kind: information flows along various channels. An *information network* is a (finite) labeled transition system, interpreted intuitively as a set of 'situations' related by some binary relations that allow flow of information from one situation to another. Concrete examples might be first-order models, with relations of isomorphism, homomorphism, submodel, etcetera.

This is a very abstract framework. What concrete questions arise? One concerns a measure for 'identity' of our notion. What is the correct structural equivalence between different information networks? *Bisimulation* seems a good candidate, just as in process theory – but this time, describing equivalent potentials for directions of information flow. Next, at least two basic logical issues suggest themselves naturally :

(1) A general calculus for combining information from different sources (regardless of the origin of its initial statements: extrinsic, intrinsic),

(2) 'intrinsic input': transfer behaviour of specific model relations.

The former is modal or *dynamic logic* (or suitable fragments of it), re-interpreted in this setting, while model-theoretic *preservation theorems* are a prime source for the latter. Thus, our starting point are the same models that underly modal process theories (Sections 2, 3). But the questions that we raise are rather different.

2 Consequence along a Connection

Information networks suggest the following key notion of 'flow' across links:

$$\mathbf{A} \rightarrow \mathbf{[R]B}: \quad \textit{if A holds in situation s and s R t, then B holds in situation t}$$

This is a generalized consequence, along such model relations as 'submodel' or 'potentially isomorphic image'. Standard consequence is the case where R is the *identity* relation. Motivation for and applications of this notion are found in Barwise & van Benthem 1996. Here is a typical result for (infinitary) first-order languages.

<u>Example</u> Bisimulation preservation and modal interpolation.
If A, B are first-order formulas, and R is *bisimulation* w.r.t. their shared vocabulary, then (1) A implies B along R iff (2) there exists a modal interpolant C such that A|= C|= B . A simple modification holds with different languages on both sides (cf. Section 10.1).

3 Complete Modal Calculi

The simplest useful inferences work as follows. Given some transfer statements A → [R]B as premises, how to derive a new one, representing some further transfer of information? What this requires is an axiomatization of the *Horn fragment of minimal polymodal logic*. (The version needed for this purpose is 'global consequence', from universal truth of the premises in a model to universal truth of the conclusion.) This is easy to do.

Richer logics to this effect use Horn fragments of dynamic logics building up complex new relations to get the right transfer statements for the conclusions. E.g., hypothetical syllogism:

$$\textit{from } A \rightarrow [R]B \textit{ and } B \rightarrow [S]C \qquad \textit{to } A \rightarrow [R\,;S]\,C$$

The *Tree Calculus* from "Dynamic Bits and Pieces" (1997) gives a concrete implementation. Its assertions generalize the schema A→[R]B to the more convenient and flexible format "description of some tree of connected models" implies "description of the root situation". This calculus was designed to describe plan formation, but it can also describe combination of information links ('planning new information'). It is reprinted in the Appendix below.

4 A First-Order Horn Clause Analysis

All the above inference can be formulated in terms of *universal Horn clauses*, whose variables range over the situations in one information network, and whose vocabulary refers to transfer relations as well as unary facts local to a situation. Horn clauses can express more sophisticated informational dependencies than what was handled above: say,

$\forall xyz\,((Rxy \,\&\, Sxz \,\&\, Tyz \,\&\, Ay \,\&\, Bz) \rightarrow Cx)$

(These richer statements are no longer preserved under bisimulation between networks.) This first-order calculus is easily decidable (a standard fact), and one complete inference system is PROLOG-style SLD resolution. Even so, there is an interest to explicit calculi for *specific* links – and expressive completeness for modal logics matching their Horn clauses.

5 Sequential and Parallel Operations

Informational inference goes in tandem with link-building. To see this, one can analyse propositional inferences with relational tags, and observe the emergence of complex links. (Cf. again the Tree Calculus of our Appendix.) Natural examples are the following:

from	$A \rightarrow [R]B,\ B \rightarrow [S]C$	*infer*	$A \rightarrow [R\,;S]\,C$	<u>composition</u>
from	$A \rightarrow [R]B$	*infer*	$\neg B \rightarrow [R^{\vee}]\neg A$	<u>converse</u>
from	$A \rightarrow [R]B,\ A \rightarrow [S]B$	*infer*	$A \rightarrow [R \cup S]\, B$	<u>union</u>

The obvious language for this is a fragment of propositional dynamic logic. But if we want to 'linearize' the two-dimensional finite action trees which arise eventually in this setting, we must use the extended Choice Calculus with main operation **&** of Section 10.6. Even then, not every propositional inference will 'fit'. We also need 'parallel' operators, as in:

from $A \rightarrow [R]B,\ C \rightarrow [S]D$ *infer* $(A, C) \rightarrow [R\mathbf{x}S]\,(B, D)$ <u>product</u>

or in first-order transcription:

$\forall xyzu{:}\ ((Ax \,\&\, Cy) \,\&\, (Rxz \,\&\, Syu)) \rightarrow (Bz \,\&\, Du)$

A concrete calculus for this purpose needs product operators on complex states in a polyadic version of propositional dynamic logic. Such modal calculi were provided in Section 2. Notice again that, although these calculi were developed to model processes (through process graphs), they also fit the current interpretation in terms of information networks.

6 Extensions with Guarded Patterns; 'Boosting'

Modal logics retain their decidability when extended to the Guarded Fragment of first-order logic (and even further; cf. Section 3). The latter allows all bounded existential quantifiers

$\exists \mathbf{y}\,(G(\mathbf{x}, \mathbf{y}) \,\&\, \phi\,(\mathbf{x}, \mathbf{y}))$

Thus, we can freely use existential modalities $\langle R \rangle$ of various kinds in our calculi (going beyond universal Horn clauses) without loss of decidability. For instance, a modal statement

$A \rightarrow \langle R \rangle\,(A \,\&\, B)$

says that A *may be boosted to* B *along* R . In modal logic, techniques like Segerberg's 'Bulldozer Theorem' or Vakarelov's 'Product Lemma' boost various properties of frames along bisimulation. (Van Benthem 1997B has more on boosting.) Also, standard *unraveling* is a construction which adds intransitivity and other tree properties along bisimulation.

7 A Complete Modal Calculus with Existence

To describe some of the previous phenomena (such as 'modal boosting') one can axiomatize the $A \to [R]B$, $A \to \langle R\rangle B$ fragment of the minimal modal logic in its own right. (The latter suffices, in a sense, for the whole system – via a well-known subformula coding trick.) But again, to get the subtler principles, one needs explicit first-order versions, too.

8 Concrete Excursion: Predicate Logic with an Extension Modality

Specific transfer facts in our richer calculi may be much more complicated than those of the form $A \to [R]B$, which were often RE (though usually not decidable). For instance, saying that A implies B along *all submodels* is quivalent, by the Los-Tarski Theorem, to stating that there exists *some universal interpolant* C such that $A \models C$ and $C \models B$. But the latter assertion is clearly RE. We discuss one similar existential case, showing how procuring base facts about 'intrinsic information flow' is highly non-trivial.

Consider first-order formulas with implications to existential modalities. These are needed, e.g., to express situation-theoretic 'constraints' like "where there is smoke, there is fire." Another motivation was the ubiquity of modal techniques like 'boosting along bisimulation'. We list some facts which are easy to prove:

Fact The general notion " $A \to \langle inclusion\rangle B$ " is not RE.

Proof One easily reduces first-order satisfiability to this notion. ■

Question What is the exact complexity of this notion?

As we shall see in a moment, the preceding implication is arithmetically definable. Our more precise conjecture is $\Pi^0{}_2$ for the relation of 'submodel'. Similar questions arise for other important model connections, in particular – with 'modal boosting' – for *bisimulation.*

Proposition The notion " $A \to \langle inclusion\rangle B$ " is equivalent to *conservativity* of A over B w.r.t. *universal* statements.

<u>Proof</u> There is a straightforward semantic argument for this. (1) First, if B implies some universal sentence C, then so does A. For, let **M** be any model for A. It has some extension **N** which is a model for B. Therefore, C holds in **N**, and by preservation under submodels, C also holds in **M**. (2) Conversely, let **M** be any model for A. Consider the atomic diagram of **M** together with B. We claim that this is finitely satisfiable. For suppose otherwise. Then B implies some negation of a conjunction of true literals in the **M**-diagram, and – quantifying out the new domain constants – we get a universal consequence of B which is false in **M**, and hence does not follow from A. This refutes the given universal conservativity. ■

Note that conservativity is typically $\Pi^0{}_2$ – which explains the earlier conjecture. By quite similar reasoning, we can determine a counterpart for 'boosting along bisimulation'.

<u>Proposition</u> The following assertions are equivalent for first-order formulas A, B :

(a) each model for A has a bisimilar model where B holds

(b) B is conservative over A with respect to *modal* consequences.

If A is a modal formula, condition (a) gives a bisimilar model where *both* A and B hold.

<u>Excursion</u> Implication up to some vocabulary

Conservativity suggests a *ternary* notion of consequence **A |= B | L** defined as follows: A *implies every L-consequence of* B . Ordinary valid consequence is A |= B | L_B, and conservative extension of A by B is B |= A | L_A & A |= B | L_A . This leads to a new calculus with ternary inferences that may also change vocabulary. E.g., A |= B | L and C |= B | L' imply $A \vee C$ |= B | $L \cap L'$. Interesting new questions arise in such a setting. E.g., does A |= B | L , A |= B | L' imply that A |= B | $L \cup L'$? The answer is: "no" in general, but "yes" for propositional logic, and suitable first-order fragments. This would provide a concrete calculus of interpolation and conservativity, beyond the usual proof systems. It also generalizes so-called 'Ramsey Eliminability' of theoretical terms in the *philosophy of science*, which turns on extension relations between theories with different vocabularies. (Historical motivation: explaining the role of theoretical terms, as opposed to observational vocabulary, in the claims made by empirical scientific theories.) Here is a negative result. Theory T^+ (vocabulary L+L') may conservatively extend theory T (vocabulary L), without every model of T having an L-bisimulation to a model of T^+ .

Another view of the matter is provided in Section 10.3. Universal or existential consequence along model relations involves modal statements across standard models. This move amounts to *evaluation of formulas both inside and across models*. In particular, an existential modality <R> shifts evaluation to some other model, suitably related to the current one. Thus we have a much more general model-theoretic

<u>Question</u> What is the complete propositional dynamic logic of the universe of models with the relations of submodel, bisimulation, and potential isomorphism (all taken w.r.t. changing vocabularies)?

In particular, might it be effectively equivalent to True Arithmetic?

9 Combined Interpolation Theorems

Let us also note that, in information networks, classical preservation results may have to be modified. For instance, suppose that we know that model **M** sits in an environment of one extension **N** where A holds, while it is a homomorphic immage of some model **K** where B holds. What is the best that we can say about **M**? Using Los-Tarski and Lyndon, one would say that **M** satisfies all positive consequences of B and all universal consequences of A. But is this also the best one can do? This may be seen as a form of generalized consequence in a three-model network with a submodel link and a homomorphism link. Indeed, we have the following generalization of the usual first-order preservation theorems:

<u>Proposition</u> If C follows at position **M** from A, B in all 3-networks as described, then there exists a *universal* consequence A' of A and a *positive* consequence B' of B such that the conjunction A' & B' implies C.

<u>Proof</u> The argument is a straightforward combination of the usual ones. Let UN(A) be the set of all universal consequences of A, and POS(B) the set of all positive consequences of B.

<u>Claim</u> $UN(A) \cup POS(B) \models C$

Let **M** be any model for this combined set. First, since **M** satisfies POS(B), the usual model-theoretic argument shows that there exists some model **K** for B, as well as a surjective homomorphism from **K** onto some elementary extension **M'** of **M**. Next, consider the atomic diagram of **M'** together with A. This set is finitely satisfiable – again by a standard argument (observing that any universal sentence true in **M'** is also true in **M**). Therefore, by the assumption of the theorem, C holds at **M'** – and therefore, it also holds at **M**.

The required conjunction A' & B' now emerges from the Claim by Compactness. ■

Obviously, since the usual model-theoretic preservation arguments 'add up' so easily here, there must be a more general combination result in the background. We leave the relevant generalisation to the reader.

10 Plans and a Resource Interpretation

The plan interpretation of process graphs makes their nodes into locations with *resources*, while relations indicate actions possibly using these resources. Intuitively, this is occurrence based (as in linear or categorial logic), and hence it leads to different notion of bisimulation, where having many successors satisfying (say) atom p is not the same as having just one. This goes beyond the framework so far, and when taken to information networks, it may require the use of ternary and general finitary relations between their nodes.

This resource interpretation requires us to resolve an ambiguity. It reads process graphs as AND trees (one has to perform all the component actions to obtain the result), not as OR trees (the usual interpretation of graphs process theories). This is the same issue that came up in our discussion of extensions for PDL: choice trees versus complex states for joint action.

11 Richer Flow Networks

In Graph Theory, networks are one major use for graphs, with basic results like the Ford & Fulkerson Theorem on maximum *flow capacity*. Can this be related to our analysis?

In probabilistic treatments (cf. Michiel van Lambalgen's work), one has *numerical* measures of *reliability* for the links. Can we extend our analysis to deal with 'quality' of transmission?

APPENDIX Resolution in Dynamic Logic as Task Calculus

Hoare Calculus is a system for proving correctness of programs, or developing correct programs. But computation is just one kind of action, and correctness assertions $\{A\}S\{B\}$ may just as well be read as descriptions of any available action S that will produce effects described by postcondition B given resources described by precondition A. Our more general planning task does not consist in proving isolated correctness statements. It is rather one of logical derivation. Given a number of routines $\{A\}S\{B\}$, how can we put together some combination of them performing some new task, from a given precondition to a given postcondition? Such a more general 'calculus of tasks' (ELD, chapter 11) is a common interpretation of propositional dynamic logic. It only involves a small fragment of the latter system, however. We can take the conditions to be Boolean, and the given actions to be atomic. So our question is, what is a complete subsystem for planning derivations?

Resolution and Monotonicity One natural method is propositional *resolution*. We can normalize task statements – by valid Disjunction of Antecedents and Conjunction of Consequents, to conjunctions of universal 'action clauses' of the forms $A \rightarrow_S B$, with A a conjunction of literals, B a disjunction, and S a program expression. We need a suitable style of reasoning on these clauses. Now, resolution is really *Monotonicity*, a very general logical inference allowing insertion of suitable formulas in syntactically 'positive' positions. E.g., consider $\neg A \vee B$, $A \vee C$. The former says A implies B. Hence, we may substitute B for A in the positive A-occurrence in the second disjunction, to get the usual resolvent $B \vee C$. This is the 'upward' view. Alternatively, we can use a 'downward monotonic' inference where $\neg C$ implied A, substituting $\neg C$ for the negative occurrence of A in the first clause. With labeled action clauses $A \rightarrow_S B$, however, some complications arise:

(**1**) First, consider analogues of standard propositional inferences. Let $A \rightarrow_S B$, $B \rightarrow_T C$. We want to conclude $A \rightarrow_{S;T} C$. What is the precise mechanism producing the right programs in these conclusions?

(**2**) Next, take action premises $A \rightarrow_S B \vee C$, $B \wedge D \rightarrow_T E$. Given that the actions separate the Boolean atoms, is there a good format for an evident conclusion at all?

We make a simple proposal based on 'plan trees' describing actions with conditions.

PlanImplications Let us replace the above correctness statements $A \rightarrow_S B$ by Boolean implications of the form $P_S A \rightarrow B$ – or more generally, by 'plan implications'

$$\Pi \rightarrow B$$

where Π describes the succesful execution of some actions from given resources, using existential modalities $P_S A$ looking *backward* into the past of the current state. In general, Π describes a finite tree of previous atomic actions, with literals true at its nodes. Thus, it may be constructed using only *literals, conjunctions and indexed program modalities* P_S. The conclusion B may be a *disjunction* of literals. As usual in Hoare Calculus, premises are universally quantified, over all available states in our model. The above examples become

(**1**) $P_S A \rightarrow B$, $P_T B \rightarrow C$, with conclusion $P_T P_S A \rightarrow C$ by downward Monotonicity. The passage to one complex program $P_{S;T} A \rightarrow C$ will come later.

(**2**) From $P_S A \rightarrow B \vee C$, $P_T (B \wedge D) \rightarrow E$, downward Monotonicity yields $P_T (\neg C \wedge P_S A \wedge D) \rightarrow E$ – or 'linearized': $P_{(A)?\,;\,S\,;\,(\neg C \wedge D)?\,;\,T} \rightarrow E$

Tree Calculus Here is a simple *Tree Calculus* justifying these inferences. Given premises of the above form, plus some tree formula Π^*, apply the following three rules.

In general, starting from $\{\Pi^*\}$, these will lead to the formation of a finite set of tree (formula)s $\{\Pi_1, ..., \Pi_k\}$, to be viewed as a *disjunction* of possible cases:

I *If the tree for some premise $\Pi \rightarrow B$ 'fits inside' some tree Π_i, at any node position, then we may write B at that node.*

II *If a tree has a disjunction D at a node, we may replace it by a disjunction of trees with the successive D-literals at that node.*

III *If a contradiction occurs at a node, remove the tree.*

A set of trees implies a disjunction B if B follows from the literals at each root. We revisit the above examples to demonstrate how this works (including the notion of 'fitting inside').

(**1**) Start: $\{P_T P_S A\}$
I: $\{P_T(P_S A \wedge B)\}$
I: $\{P_T(P_S A \wedge B) \wedge C\}$
The literal C at the root implies the desired conclusion.

(**2**) Start: $\{P_T(\neg C \wedge P_S A \wedge D)\}$.
I: $\{P_T(\neg C \wedge P_S A \wedge (B \vee C) \wedge D)\}$
II: $\{P_T(\neg C \wedge P_S A \wedge B \wedge D), P_T(\neg C \wedge P_S A \wedge C \wedge D)\}$
III: $\{P_T(\neg C \wedge P_S A \wedge B \wedge D)\}$
I: $\{E \wedge P_T(\neg C \wedge P_S A \wedge B \wedge D)\}$
The desired conclusion E follows from inspection of the root.

<u>Theorem</u> The Tree Calculus is complete for our task inference.

<u>Proof</u> Starting with set $\{\Pi\}$ for the conclusion $\Pi \rightarrow B$, perform all possible inferences in the calculus, using the given premises to perform substitutions. Remove trees which are subtrees of other ones. (These are implied.) This process will stop after finitely many steps. It only produces trees richer than the original one – which therefore imply it, in an obvious sense. Now, suppose some tree Π_i in the resulting set has root literals whose conjunction fails to imply B. It gives a counter-model to the implication as follows. Take Π_i *itself* as a model, with only the atomic relations described, and only those atomic propositions true at each node that are explicitly indicated at it. Evidently, B fails at the root. But, each premise is true at every node in this model. For, if its antecedent tree is true at a node, then it 'fits' inside Π_i (this is because of the special form of the corresponding modal formulas), and hence, it would have given rise to a further I-move adding literals. In general, this will be a disjunction, whence a further II-move was applied, yielding trees with extra literals (as compared with Π_i). Not all of these can have been removed by III-moves, or Π_i would not have made it into the final set. But the other situations are impossible, too, as Π_i would

then have been removed for not being maximal. The outcome must be that no antecedent of a premise is true at any node in our model – and hence all premises hold vacuously. ■

A complete calculus of task inference is no surprise. Inference between plan implications is *decidable*, even with premises read universally (ELD, Chapter 7, Theorem 10).

Program Operations for Hoare-Style Conclusions Is there a standard procedure for *linearizing* statements $\Pi \to B$ into standard correctness assertions $A \to_S B$, of course, for suitable complex programs S ? The matter is not entirely clear. Branching tree patterns call for *parallel* program operators, going beyond dynamic logic. E.g., premises $A \to_S B$, $C \to_T D$ suggest a conclusion $A \wedge C \to_U B \wedge D$ for some new program U . One option for U might be Boolean intersection $S \cap T$. But we can also use new parallel operators. Tree transcription of our premises suggests a conclusion $(P_S A \wedge P_T C) \to B \wedge D$, whose linearisation might read $\underline{true} \to_{((A)?\,;\,S)\,||\,((B)?\,;\,T)} C$. A third option are n–ary modalities directly over tree-like structures (cf. Hollenberg 1998), that support parallel programs. The design of a suitably expressive repertoire of program operations for our task calculus remains open. But then, *trees themselves* may be just as convenient representations of plans.

Synthesizing Plans The Tree Calculus also helps synthesize plans out of premise routines. Now, we have 'resource propositions' $\mathbf{A}$ and a 'goal' G , and a 'plan' is a tree with leaves from $\mathbf{A}$ only which implies G. One procedure enumerates all possible resource-to-goal implications from the given premises (with their plan trees). A finite upper bound to the number of the latter can be computed in advance (it only depends on the proposition letters occurring in the problem). Then, we solve the standard propositional search problem from $\mathbf{A}$ to G using the derived implications. An associated plan with intermediate actions arises from successive leaf substitution of trees for auxiliary implications.

<u>Example</u>

Let the resource proposition be A and the goal G . The available action premises are $P_S B \wedge C \to G$, $P_T B \to C$, $P_U A \to B$. We derive G from A as follows:

1	G from B, C
2	B from A
3	C from B
4	B from A

The associated trees will work out to (via their above normal form descriptions):

1	$P_S B \wedge C$
2	$P_S P_U A \wedge C$

3 $P_S P_U A \wedge P_T B$

4 $P_S P_U A \wedge P_T P_U A$ ■

Less blindly, we need a *search procedure* for finding good conclusions (including plans). Now, notice that the preceding example looks somewhat like a logic program derivation. Here we need a translated first-order version of our plan implications, in the standard modal fashion. Consider the earlier Example (**1**). Take first-order clause forms for its two premises: $Ax \wedge Sxy \rightarrow By$ and $Bx \wedge Txy \rightarrow Cy$. From an assumption Au , the standard search procedure for a proof of the goal Cv will produce outcome $Sus \wedge Tsv$ – whose quantified version $\exists s\,(Sus \wedge Tsv)$ is exactly the definition of program composition proposed earlier. The preceding example may be analyzed in a similar manner through its first-order transcriptions, trying to get Gv from instances of Au using the clauses

$Bx \wedge Sxy \wedge Cy \rightarrow Gy$ $\quad$ $Bx \wedge Txy \rightarrow Cy$ $\quad$ $Ax \wedge Uxy \rightarrow By$

Thus, standard proof search via first-order transcriptions may produce useable answers.

Another angle on plan synthesis is 'propositional completeness'. All valid consequences between plan implications reduce to valid propositional inferences by disregarding all action operators P_S . (These consequences must also hold on models where all atomic relations coincide with the identity relation.) Conversely, consider any valid propositional inference from a set of implicational clauses to one implicational clause $D \rightarrow E$. Now, assume that the premise clauses all carry an action S producing their consequent from their antecedent.

<u>Question</u> Is there always a plan implication $\Pi \rightarrow E$ for a valid conclusion whose antecedent Π only employs conditions that occur in D ?

A positive answer would express a kind of functional completeness for the programming repertoire encoded in our Tree Calculus. Finally, we mention a case of plan inference where additional expressive power seems needed.

Negations and Converse The obvious dynamic version of propositional Contraposition

$A \rightarrow B \models \neg B \rightarrow \neg A$

is the inference from

from $P_S A \rightarrow B$ *to* $P_{S^\smile} \neg B \rightarrow \neg A$

involving a *relational converse* $S^\smile$. Contraposed once more, this implication reflects the well-known tense-logical 'duality inference' *from* $P\,A \rightarrow B$ *to* $A \rightarrow G\,B$. This example shows that we need plan trees which also allow converse arrows, going to successors, rather than predecessors in the atomic relations. It may be checked that the above rules remain complete. E.g., dynamic contraposition remains derivable in this fashion.

9 Information Processing as a Social Activity

Colloquim on *Social Organisation in Logical Theory*, TU Eindhoven, March 1998. The following are points from an abstract for a talk, together with some observations prompted by a day of pleasant discussion at Eindhoven Technical University.

1 Logic in Groups Traditional logic is mainly about single agents that think, reason and evaluate. But social themes are emerging nowadays. Our somewhat Pickwickian sense of 'social' themes employed here: *all those issues where a group level is essential*.

2 Epistemic Logic A famous case where a social level leads to significant logical insight is Epistemic Logic, in its gradual development from individual knowledge to *group knowledge*. Hintikka talked about single agents which can reflect on each other's information through finite iteration of knowledge operators K_i, K_j . Lewis put 'common knowledge' on the map in his study of conventions and rules, R. Fagin, J. Halpern, Y. Moses & M. Vardi 1995 has a full-fledged theory of 'collective epistemic operators' E_G ("everyone in group G knows"), C_G ("common knowledge in G "), IG ("implicit knowledge in group G"). Common knowledge is a typical group phenomenon (what is known in 'reflective equilibrium'), as is implicit knowledge (what is known by pooling the individual information). No explicit calculus of groups occurs in epistemic logic, which would take this emancipation of social structure one step further.

Questions

Introduce groups as an explicit object of study, in a dynamic logic with manipulation of G-arguments, not just proposition arguments. An example is a modal calculus of social combination inferences such as (1) $C_G A$ implies $C_{G'} A$ for all subgroups G' of G (valid for factual propositions, invalid for statements of ignorance), or (2) combinations of group knowledge, such as $(C_{G1}A \ \& \ C_{G2}B) \rightarrow C_{(G1 \cup G2)}(A\&B)$ (invalid), or $(I_{G1}A \ \& \ I_{G2}B) \rightarrow I_{(G1 \cup G2)}(A\&B)$ (valid). This calls for systematic comparison with dynamic logics and process algebras for parallel computation. (Common knowledge can be viewed as referring to a program $(i_1 \cup ... \cup i_k)^*$ where $G = \{i_1, ..., i_k\}$. What is the natural group structure allowing for cooperation between subgroups?

3 Reducible versus emergent group properties. E reduces to properties of individuals, C 'half' (in a circular manner), some things not at all. Compare the semantics of *collective predicates* in natural language, which is notoriously hard. E.g., the meaning of a simple, almost 'logical' expression like the reciprocal "each other", turning an individual predicate into a collective one applying to groups, has been under debate for decades. (No one has such difficulties with its individual cousin "self"...) Similar problems afflict plural quantifiers (van der Does 1992). This is a serious issue. Perhaps the collective talk pervading communication in natural language has no definite truth conditions at all, only partial constraints! If "the prisoners liberated each other", some prisoner liberated some prisoner. There may not be more 'regularity' than that, though by no means everyone need have liberated everyone. And as every academic knows, if "the professors quarrelled" it is even less clear what happened.

<u>Questions</u>

> Study many-level languages mixing knowledge and action of both individuals and collectives, allowing for some reduction between levels as the case may be.

4 Semantics of Communication Language use and reasoning is a social proces. Contemporary logical semantics is moving from its original habitat of single sentence meanings towards *discourse and communication*. How to deal with these social phenomena without losing the subtlety and rigour that has been achieved lower down? One concrete challenge in this move is one single building block of dialogue, the communicative unit consisting of a *question/answer* exchange. This crucially affects collective information states of questioner and answerer, by suitable updates for the two speech acts. This is an active research area, witness Jaspars 1994), Gerbrandy & Groeneveld 1997, as well as recent research by Jeroen Groenendijk, dynamifying Groenendijk & Stokhof 1984.

<u>Questions</u>

> How to model collective communication states, and important updates? More generally, how to take communicative actions like questions seriously as a new category in logical theory – in addition to proof steps or evaluation moves?

<u>Excursion</u> *Specifying preconditions/postconditions, or specifying updates directly?*
In this area, two logical approaches occur which are interestingly different. The Bunt–Jaspars line specifies the relevant dynamic process in terms of preconditions and postconditions. Thus, a question-answer exchange between agents Q and A might be any move which starts from Q-ignorance about some proposition P and Q-knowledge that A knows if P, to a state where it is common knowledge that both know if P. The exact nature of the update can be left open. Conversely, in much Amsterdam work, information states and their updates are central (satisfying key intuitions) – after which one will just have to see if they satisfy the relevant postconditions. E.g., Gerbrandy's 'Dynamic Epistemic Logic' has an update operator **learn**(P) when agent i learns that P . This changes i's information state, updating all his alternatives

with P, while leaving the alternatives for all other agents unchanged. (This is hard to implement over ordinaryKripke models, which generate 'side-effects' for i's update, affecting the others' knowledge after all. To avoid this, Gerbrandy uses non-well-founded sets.) Thus, the intuition here is some form of minimal change. Can this also be cashed out in terms of pre- and postconditions? After the update, i has 'only' learnt that P, while the others have not learnt anything new at all. In dynamic logic terms, the postcondition should be something like the backward-looking converse modality: SP(A, **learn**(P)) := $\langle$**learn**(p)$^{-1}\rangle$A. But this statement is undefinable in the usual update languages. Connections between update systems and pre/ postcondition specifications in static epistemic languages for group knowledge are still scarce.

5 Game Theory The oldest social paradigm in logic are *games*, that go back to Antiquity. Paradigmatic modern examples are Hintikka evaluation games, Ehrenfeucht comparison games, and richest of all for analysing communication: Lorenzen argumentation games. Up until now, logical games have mainly served to throw new light on existing notions. But they embody many ideas that are sui generis, such as *commitment, role, role switch, strategy, game resources, 'social construction' of a common object*. Games are on the way up in logic, as a means of exploring new avenues. (Compare the recent work on games for linear logics.) Moreover, there are some interesting junctions between Game Theory in the received sense of that term and epistemic logic, e.g., in the work of Bonanno and Vilks. (Cf. Dekel & Gul 1997.)

Questions

> What is a paradigm for 'logical games' comparable in scope to the received analysis of formal proof, or formal computation? Who will solve the meta-equation *? : game = Hilbert : proof = Turing : computation* ? How to import probabilistic considerations (at the heart of classical Game Theory) into logical games? What are probabilistic moves – or on another line, how could one certify, without playing all possible games, with sufficiently high probability, my possession of a winning strategy in logical games?

6 Many-authored Theories 'Social themes' in logic correlate with developments in the philosophy of science. First, consider information *representation*. The 'web of scientific theories' is group knowledge of a whole field. Since the Renaissance, no single individual's state contains this. Moreover, there are several questions about its architecture. One is aggregation: possible consistency problems when merging theories. The other is segregation: how to encapsule parts of theories in a modular fashion, so that failures in one module need not vitiate the whole? Relevant logical work may be found in the literature on 'combining systems', as advocated by Gabbay. There are interesting analogies between work on theory structure, and the structuring of information states in semantics of conditionals and epistemic updates (cf. Segerberg's recent work on so-called 'hyper-theories', and the discussion in Subsection 10.4).

Questions

Give a calculus of social knowledge architecture, with natural inter-theory relations and combinations. How to combine this with current logics of belief update and revision? And with preference structure in default logic?

7 Representation and Computation Representation invites computation. Cognitive action is also becoming a central theme in the philosophy of science. This started with Popper's pioneering emphasis on *learning* as a basic category – a theme which is also slowly penetrating into logic and computer science. It is quite explicit in Theo Kuipers' recent broad monograph on *Cognitive Structures in Science*, Philosophical Institute, Rijksuniversiteit Groningen. Social processes (in our logical sense) in science involve: argumentation games, the role of 'the forum', betting models for rationality (which involve several players), collective aspects in scientific proof, theory change, language change, etcetera. Again, these lead to interesting analogies with developments elsewhere (such as logic, or Artificial Inteligence); cf. Aliseda-Llera 1997.

Questions

Analyse classical problems in the philosophy of science in logical dynamics for 'social' structures. Compare specific themes in logic and philosophy of science. E.g. key notions of verisimilitude and truthlikeness in Zwart 1998 resemble those found in AGM-style belief revision theory. Theory structure often has a syntactic flavour. Thus, how can one translate systematically between epistemic logic and syntactic proofs: K_iA and 'i has a proof for A' – individually, or socially?

8 Conclusions A social aspect is emerging in current logical studies. There is even more evidence for this claim than what we have surveyed here, such as interesting analogies between dynamic epistemic logic and the key phenomena studied in *Social Choice Theory* (cf. the introduction in the recent logic textbook by Royakkers and Sarlemijn). But the agenda and paradigm for the study of social, collective structure in logical terms are still unclear. What this move leads to is an interesting generalisation of logic. Not just individuals can have goals and transform information. So can social organisations, which are epistemic agents just as individuals. A major challenge, therefore, to logic as classically conceived, is extension of its scope so as to deal with information flow in significant organisations.

10 Paralipomena

This final section collects some disconnected fall-out of the preceding investigations.

1 Bisimulation Invariance and Translation

The following point was made by Natasha Kurtonina. Intuitively, 'simulations' may relate processes with *different* moves or local properties. But then, the usual model theory of bisimulation – for instance, as presented in ELD – is too uniform, as it has the same language on both sides. Here is a first response.

Consider two modal models **M**, **N**, in different similarity types L, L'. A bisimulation is a binary relation E between points in the two models with the following properties:

(i) there exists a correlation between L-atoms p and L'-atoms q such that if s E t, and **M**, s |= p, then **N**, t |= q ; and vice versa

(ii) there exists a correlation between L-actions a and L'-actions b such that if s E t, and s a s', then there exists t' with t b t' and s' E t'; and v.v.

This corresponds to a fixed correlation of features observed in one process with those in another. In this case, each L-formula ϕ has a direct L'-translation $\tau(\phi)$ (and vice versa).

Theorem The following assertions are equivalent:

(a) ϕ implies ψ along L-L'-bisimulations

(b) there exists some modal L-formula α such that $\phi \models \alpha$, $\tau(\alpha) \models \psi$

The proof is essentially the argument for the Modal Invariance Theorem. From (a) to (b), one shows that the set of all L'-translations of the modal L-consequences of ϕ implies ψ.

But there are further natural situations. Suppose we have a more complex correlation, with an occurrence of p in **M** corresponding to truth in **N** of some complex L'-formula σ_p , and the occurrence of an a-move in **M** always matched by some finite sequence of actions in **N** defined by some expressen σ_a . Assume the same in the opposite direction, with a similar translation τ. Then we need a more complex two-way preservation statement. E.g., immediately preserved from left to right under σ-translation are all formulas generated by the syntactic schema $p \mid \neg p \mid \& \mid \vee \mid \langle a \rangle$. But in addition, universal modalities $[\tau(b)]$ may be allowed, when translated into plain L'-modalities [b].

<u>Question</u> What is the proper treatment of the preceding situation?

Things get even more complicated if we want the definability to come out, not by fiat, but as a result of some semantic regularities – as happens in Beth's Definability Theorem.

2 How to Express Variable Dependencies

Decidable remodelling of first-order logic can be done in the form of generalized assignment models, where 'gaps' encode *dependencies* between variables. Now dependencies are interesting mathematical structures in their own right. But are they adequately reflected in the standard predicate-logical language? We give some examples suggesting the need for, at least, an enriched modal logic on top of the latter.

Consider the main example of a generalized assignment model in ELD, chapter 10. It has a domain of objects {1, 2}, a set of variables {x, y}, and so there are 4 possible states. These generate 15 non-empty assignment models, which may encode various *dependencies* between the variables. For instance, the one with just {(x, 1), (y, 2)} and {(x, 2), (y, 1)} made y and x heavily interdependent: a change of value for one forces a change for the other. Now we showed how to interpret a predicate-logical language with quantifiers ∃x, ∃y over all these models. But is this really the right medium for bringing out the underlying dependencies, viewed as important structures in their own right?

Let us look at the situation in modal logic. The standard model is really a 4-world *multi-S5 model* with two modalities, which may be drawn as follows:

		xy 11		
	x		y	
xy 21				xy 12
	y		x	
		xy 22		

To liven things up, we can postulate some binary relation R on the underlying objects, say, R = {(1, 2)}. Let the language contain atoms Rxy, Ryx. Each of these will be true in one world in the above picture. One way of making distinctions between dependency models is by looking at all possible submodels of this multi-S5 model, and asking if their modal theories are different. It is possible to show that they are, by inspecting all cases. But this does not define dependency information *directly*. And indeed, standard predicate logic seems to poor to adequately describe, say, the above 2-world model. No ordinary relation $=_x$ or $=_y$ crosses from one world to the other: only $=_{\{x, y\}}$ does that.

More generally, we need an extended modal logic with modalities for all relations

$w =_X v$ iff *w, v agree on all variables except at most those in the set X*

But will this express concrete facts about variable dependencies? Consider two examples:

(a) "If x changes its value, than so does y"
(b) "Any change in x determines a unique change in y"

Neither assertion is expressible in even a polyadic quantifier language, although some approximations may be stated. But these are contrived and indirect. It seems we need further relations $=^X$ which say that, in passing from assignment w to v *at least* (not: 'at most') the variables in X change their values. Then we can express (a) as follows:

$$<=^{\{x\}}>\phi \rightarrow <=^{\{x, y\}}>\phi$$

This is another case where generalized semantics supports natural new types of quantifier, beyond the standard first-order ones. The above type of quantification seems related to introducing some kind of *difference modality* between states.

<u>Questions</u> What happens to *decidability* and *axiomatization* of generalized predicate logic when we add a difference modality – or even just a universal modality? Does this correspond to an obvious extension of the Guarded Fragment?

Here is an almost–translation into the Guarded Fragment with identity. Let ϕ have variables $\{x_1, ..., x_k\}$ in total, and let y be a new variable, different from these and x:

$$<=^{\{x\}}>\phi \Leftrightarrow \exists x_1...x_k\, y. (R(x_1, ..., x_k, y)\ \&\ y{\neq}x\ \&\ \phi(y/x)(x_1, ..., x_k))$$

where R is the uniform relativizing predicate for all quantifiers used in Andréka, van Benthem & Németi 1998 in order to reduce satisfiability in generalized assignment semantics to standard satisfiability in GF.

But note that this introduces new variables, and does not seem to do the job precisely.

Taking dependence models seriously means finding the right modal language for them – and then developing its simulations, correspondences, and complexity properties.

3 Lowering Complexity by Long-Distance Evaluation

Various strategies for lowering complexity occur in logical dynamics. One is the use of 'general models' for second-order logic, restricting predicate ranges. Another is algebraic relativization, restricting available object combinations. But one can also vary the mechanism of the truth definition for similar purposes. We consider evaluation allowing jumps across 'indistinguishable' models as one further strategic remodeling option, with some good independent motivation, and raise some questions about its effects.

There are few general strategies for lowering the complexity of logical systems. One is the use of Henkin's *general models*, which turn non-arithmetical second-order logics into RE many-sorted first-order logics. Another is algebraic *relativization*, which turns RE but undecidable algebraic logics into decidable ones. But here is another approach, inspired by the discussion of modal logic with 'bisimulation quantifiers' $B\phi$ in Hollenberg 1998, which 'jump models' by stating that ϕ holds in some bisimilar state in a possibly different model. These quantifiers access the current model only 'up to bisimulation'.

Here is our proposal for second-order logic. The problem with predicate quantifiers $\exists Y$ is their ranging over the power set of the current model **M**, a mysterious set-theoretic entity. Let us allow these quantifiers to be a bit fuzzier now, claiming the existence of a set that we know 'up to a degree' measured by some semantic equivalence relation. In general, formulas will have free object variables **x** set to objects **a**, and free predicate variables **X** set to predicates **P**. Here is a new second-order quantifier clause:

$\mathbf{M, a, P} \models \exists Y \bullet \phi$ iff *there is a model* ***N, b, Q*** *potentially isomorphic to* ***M****, and a set B in* ***N*** *such that N, B* $\models \phi$

The relevant potential isomorphism generalizes that of first-order logic. Its component partial isomorphisms refer to predicates in **P** concerning **a**-objects, matching their counterparts in **b** w.r.t the corresponding **Q**-predicates – plus the constant predicates of the language. This move does not make a difference over countable models, as potential isomorphism is isomorphism there, but it does when we work on arbitrary models.

<u>Question</u> What are the complexity effects of this move?

For independent motivation, cf. 'consequence along a model relation R' (Barwise & van Benthem 1996), with a modal form $\phi \to [R]\psi$. Our $\exists Y$ is an existential modality <R>.

4 Towards a Dynamic Theory Structure

Constructive information states can be thought of as 'theories', in the sense used in the philosophy of science. We identify a number of stages where theory structure is becoming more complicated these days, plus some analogies with the needs of dynamic semantics. These thoughts were inspired by Zwart 1998.

Verisimilitude is a ternary relation $V_A BC$ saying, intuitively: 'B is more like A than C is'. To some extent this may be compared with a notion of (preferential) *consequence* from C to B, in the context of C. Sjoerd Zwart's recent dissertation surveys many proposals for more precise definitions, constraints on how the latter are to perform, and in the process, different representations of the 'theories' involved in this comparison. Here are some analogies with issues in logic.

4.1 Base Level Here is a first view of theories. T is a set of sentences, which corresponds semantically to MOD(T), the class of all models that verify every sentence in T. Tarskian consequence operates at this level:

$$T_1 \models T_2 \quad \text{if} \quad MOD(T_1) \subseteq MOD(T_2)$$

4.2 Partial Logic Let a theory now consist of two disjoint classes of models: $MOD^+(T)$, the ones that are definitely accepted, and $MOD^-(T)$, the ones that are definitely rejected. The remaining models form a grey zone. This is exactly as in 3-valued logic, and consequence becomes a bit les clear-cut accordingly. Here are two options:

$$T_1 \models^+ T_2 \quad \text{if} \quad MOD^+(T_1) \subseteq MOD^+(T_2)$$
$$T_1 \models^{+-} T_2 \quad \text{if} \quad MOD^+(T_1) \subseteq MOD^+(T_2) \text{ and } MOD^-(T_2) \subseteq MOD^-(T_1)$$

This emergence of options for defining logical consequence may match the well-known proliferation of options for verisimilitude. No unique best choice may exist.

4.3 Hypertheories Now lift theories to *families of sets* of models (there are motivations for this in linguistics and AI). E.g., think of the family $\{ MOD(\phi) \mid \phi \in T\}$. This is intermediate between making theories syntax-independent and syntax-dependent. E.g., {p, p &q} will be different from {p&q}, but the same as {p&p, q&p}. Valid consequence between theories at this level is even a less clear-cut intuitive notion. Should one require, perhaps, that

$$\forall X \in T_2 \; \exists Y \in T_1 \;\; Y \subseteq X \; ?$$

<u>A Difficulty</u>. There are two interpretations for this. *Conjunctive*: the theory says that all models in its intersection are 'in', while those outside of all sets of the family are 'out'. The family records how the intersection was arrived out, as a handle for later belief revision, or other cognitive processes. *Disjunctive*: the theory says that one of the sets in the family is the right one. In this case, the intersection records what is 'in' no matter what, and the exterior everything that is 'out'. Sjoerd Zwart's 'modal theory representation' in terms of S5 normal forms is of the second variety: it describes all S5 models in which the theory would be true.

<u>Question</u> What is the connection between this view of verisimilitude and hypertheories for belief revision as developed by Krister Segerberg?

Of course, there are even richer theory representations, indicating preferences between different pieces, as in Mark Ryan's well-known dissertation on 'structured theories'.

5 Updates, Upgrades, and Setting an Agenda

Incoming assertions need not just increase information, say, by eliminating possibilities. They may also change current preferences over these possibilities (as being 'more or less plausible'), or they may merely structure the set of assertions now on the table. We present a simple propositional model for doing this.

Model 1: Updates Information states are sets of propositional valuations. <u>Update(ϕ)</u> is an instruction (alternatively, a mode of reading the incoming assertion) which eliminates all valuations that do not verify ϕ, viewed as a classical proposition.

Model 2: Updates and upgrades Information states are now 'graded' sets of propositional valuations, where each valuation has a natural number indicating its 'current preference status'. Update(ϕ) works as before. <u>Upgrade(ϕ)</u> adds 1 to each valuation which verifies ϕ in the standard sense.

Model 3: Updates, upgrades, and tabling Information states are graded sets of propositional valuations, plus a marking of subsets named by specific formulas ('what's on the table'). Update (ϕ), Upgrade(ϕ) work as before. <u>Table(ϕ)</u> adds a marking to the current table for the set of valuations verifying ϕ.

The final model is a bit like the hypertheories of an Section 10.4, as it carries 'historical' information. The upshot of all this is a rich procedural version of propositional logic, which can be used as a concrete model for studying issues like

(1) laws of felicitous discourse: 'no update befor tabling', etcetera
(2) logic of discourse moves: such as recursion rules for Update, Upgrade, and Table w.r.t the standard Boolean connectives
(3) new procedural notions of validity, or other items of importance to argumentative discourse.

References
J. van Benthem, J. van Eyck & A. Frolova, 'Changing Preferences', CWI, 1995.
F. Veltman, 'Defaults in Update Semantics', *Journal of Philosophical Logic*, 1996.

6 Choice Trees in Dynamic Logic

Labeled transition systems are a *disjunctive* definition of all possible steps in a process. Standard dynamic logics do not manipulate such choice trees. We briefly sketch a modal extension of PDL which does.

6.1 Trees and Process Graphs *Choice tree*: finite graph with arrows for actions (perhaps including tests). OR-interpretation: the various options of a single process. Distinguish from AND interpretation: joint action (as in Section 2). Intuitive ambiguity "and"/"or" interpretation: cf. the deontic 'Paradox of Free Choice Permission'.

6.2 Language and Semantics Language. $\langle G\rangle\Phi$. We record the nodes of the tree G for use in the syntax, while Φ is an assignment of formulas to these nodes. Example: single-tree equivalent for $\langle a\rangle\phi \wedge \langle b\rangle\psi$ with branching tree $\langle a + b \rangle \langle\phi, \psi\rangle$. Interpretation in standard PDL models: via existence of a succesful embedding of G into the model, starting from the current state as its root.

<u>Fact</u> Every tree-formula is equivalent to an ordinary PDL-formula.

Reason: trees can be successively 'unpacked' by conjunction of options plus composition for continued branches. Next, consider *tree operations* **&** and $\bullet$. The first adds trees under a joint root ('choice'). The second glues a tree under another at some specified leaf, for 'continuation' of processes. (Options: glue at any node, or at specified leaves only.)

<u>Fact</u> Initial **&** and final $\bullet$ are complete for building all finite trees.

6.3 Axioms and Completeness Distribution laws for **&** and $\bullet$ describe equivalent ways of constructing a tree. They result in a 'normal form' description which belongs to the original PDL. This is also the complete axiomatization. Also reflected in the logic: differences between **&** and program union $\cup$.

6.4 Iteration and Fixed Points Implicit definitions and *iteration*. The outcome becomes really stronger than PDL. Example: fixed point for the tree matrix

{q, a-b branch to <*, *>}.

Solution: all finite trees in which every node is either a p-leaf, or it has both an a- and a b-successor that each start a similar tree.

<u>Fact</u> This class is undefinable in PDL, which defines only regular languages.

Nevertheless, this extended language is still *decidable*.

<u>Fact</u> All tree fixed points are definable in the μ–calculus.

<u>Example</u> The above statement about binary trees is $\mu q \bullet\ p \vee (\langle a\rangle q \wedge \langle b\rangle q)$

Complete axiomatization? The two obvious valid iteration principles reflect properties of 'smallest pre-fixed point':

(1) $\phi\ (\mu q \bullet \phi(q))$ (2) if $\phi(\alpha) \to \alpha$, then $\mu q \bullet \phi(q) \to \alpha$

General analysis: effective translation into (a small recursive fragment of – countably) *infinitary modal logic*: the above fixed points use only very simple countable disjunctions. Generalisation of Kleene's Normal Form Theorem for regular expressions: tree notations.

Most striking feature: all relevant fixed points are reached after ω approximation steps, because the associated operators are *finitely distributive*. Syntactic normal form for such special operators: $\mu q \bullet \phi(q)$ where the occurrences of q lie only in the scope of $\vee, \wedge, \exists$. This is a tree-style generalisation of Kleene's syntactic regular notation.

6.5 Invariance and Safety The extended language (including all propositional fixed point operations) is invariant for bisimulation.

<u>Proposition</u> Safety for the new tree operations follows by an easy induction.

Converse: finitary and infinitary versions exist (cf. Barwise & van Benthem 1996), but we have the same difficulty as ever in zooming in more precisely on just fixed point logic.

References

A. Aliseda-Llera, 1997, *Seeking Explanations: Abduction in Logic, Philosophy, and AI*, Ph.D. thesis, Symbolic Systems program, Stanford University & ILLC.

H. Andréka, J. van Benthem & I. Németi, 1998, 'Modal Logics and Bounded Fragments of Predicate Logic', *Journal of Philosophical Logic* 27:3, 1998, 217–274.

J. Barwise & J. van Benthem, 1996, 'Interpolation, Preservation, and Pebble Games', Report ML–96–12, Institute for Logic, Language and Computation, Amsterdam. To appear in the *Journal of Symbolic Logic.*

J. Barwise & L. Moss, 1996, *Vicious Circles. On the mathematics of non-wellfounded phenomena*, CSLI Publications, Stanford.

J. Barwise & J. Seligman, *Information Flow*, Cambridge University Press, 1997

J. van Benthem, 1985, *Modal Logic and Classical Logic*, Bibliopolis, Napoli.

J. van Benthem, 1986, *Essays in Logical Semantics*, Reidel, Dordrecht.

J. van Benthem, 1996, *Exploring Logical Dynamics*, CSLI Publications, Stanford.

J. van Benthem 1997A, 'Dynamic Bits and Pieces', Report LP-97-01, Institute for Logic, Language and Computation, University of Amsterdam.

J. van Benthem 1997B, 'The Range of Modal Logic', Report ML–97–05, Institute for Logic, Language and Computation, University of Amsterdam. To appear in the *Journal of Applied Non-Classical Logics*, memorial issue for George Gargov.

J. van Benthem & J. Bergstra, 1995, 'Logic of Transition Systems', *Journal of Logic, Language and Information* 3:4, 247–283.

J. van Benthem, J. van Eyck & A. Frolova, 'Changing Preferences', CWI, 1995.

P. Blackburn, M. de Rijke & Y. Venema, *Modal Logic*, textbook, ILLC Amsterdam & computer linguistics, Saarbruecken.

E. Börger, E. Grädel & Y. Gurevich, 1996, *The Classical Decision Problem*, Springer, Berlin.

C. Butz & I. Moerdijk, 1997, 'A Definability Theorem for First-Order Logic', to appear in the *Journal of Symbolic Logic.*

C.C. Chang & H.J. Keisler, 1970, *Model Theory*, North-Holland, Amsterdam.

E. Dekel & F. Gul, 'Rationality and Knowledge in Game Theory', in D.M. Kreps and K.F. Wallis, eds., *Advances in Economics and Econometrics: Theory and Applications*, Vol. 1, Cambridge University Press, 1997.

J. van der Does, 1992, *Applied Quantifier Logics. Collectives and Naked Infinitives*, dissertation, ILLC, University of Amsterdam.

H.C. Doets, 1996, *Basic Model Theory*, CSLI Publications, Stanford.

R. Fagin, J. Halpern, Y. Moses & M. Vardi, 1995, *Reasoning about Knowledge*, MIT Press, Cambridge (Mass.).

J. Gerbrandy & W. Groeneveld, 1997, 'Reasoning About Information Change', *Journal of Logic, Language and Information* 6:2, 147–169.

E. Grädel, 1997, 'On the Complexity of the Guarded Fragment', preprint, Department of Informatics and Mathematics, RWTH Aachen.

J. Groenendijk & M. Stokhof, 1984, *The Semantics of Questions and the Pragmatics of Answers*, dissertation, Filosofisch Instituut, University of Amsterdam.

J. Groenendijk & M. Stokhof, 1991, 'Dynamic Predicate Logic', *Linguistics & Philosophy* 14, 39 100.

M. Hollenberg, 1998, *Logic and Bisimulation*, Ph.D. Thesis, Philosophical Institute, Utrecht.

D. Israel & J. Perry, 1991, 'What is Information?', Report CSLI-91-145, Center for the Study of Language and Information, Stanford University.

J. Jaspars, 1994, *Calculi for Constructive Communication*, dissertation, ITK Tilburg & ILLC Amsterdam.

H. Kamp & J. van Eyck, 1997, 'Representing Discourse in Context', in J. van Benthem & A. ter Meulen, eds., 179–237.

M. Marx, 1997. 'Complexity of Modal Logics of Relations', Report ILLC-ML-97-02, Institute for Logic, Language and Computation, University of Amsterdam.

M. Marx & Y. Venema, 1996, *Multi-Dimensional Modal Logic*, Kluwer, Dordrecht.

M. de Rijke, 1993, *Extending Modal Logics*, Ph.D. Thesis, ILLC, University of Amsterdam.

E. Spaan, 1993, *Complexity of Modal Logics*, dissertation, Institute for Logic, Language and Computation, University of Amsterdam.

A. Tarski, 1986, 'What Are Logical Notions?', in J. Corcoran, ed., *History and Philosophy of Logic* 7, 143-154.

F. Veltman, 1996, 'Defaults in Update Semantics', *Journal of Philosophical Logic* 25, 221-261

S. Zwart, 1998, *Approaching the Truth*, dissertation, Filosofisch Instituut Groningen and ILLC Amsterdam.