Relational Validity & Dynamic Predicate Logic

Albert Visser

Department of Philosophy, Utrecht University

Heidelberglaan 8, 3584 CS Utrecht, The Netherlands

email: Albert.Visser@phil.ruu.nl

Abstract

In this paper we prove that the principles in the language with relation composition and dynamic implication, valid for all binary relations, are the same ones as the principles valid when we restrict ourselves to *DPL*-relations, i.e., relations generated from conditions (tests) and resettings.

1 Introduction

Consider a domain, i.e., a non-empty set of objects D. Consider some set $\mathcal{R}$ of binary relations over D. There are two operations on relations, we want to focus on. The first is ordinary relation composition and the second is dynamic implication (to be defined below). We close our relations under these operations. The relations so obtained, together with the operations form a structure. Here we count the relations of our original set $\mathcal{R}$ as 'constants'. We call such structures *mDIL*-models. The sequents valid for *mDIL*-models were studied by Patrick Blackburn and Yde Venema in their [3]. Now, suppose our domains are sets of assignments, functions from some specified set of variables to some given non-empty set E, i.e. $D = E^{Var}$, and let's restrict our set of basic relations to relations that are generated by composition from *finitely restricted tests* (or: *finitely restricted conditions*) and *random resettings* of single variables. Here a *test* or *condition* is a subrelation C of the identity relation —in other words: a relation without 'dynamic powers'. A test is *finitely restricted* if there is a finite set of variables W such that:

$$fCf \text{ and } f{\restriction}W = g{\restriction}W \Rightarrow gCg$$

We often confuse C with the set $\{g{\in}D^{Var} \mid gCg\}$. Thus, the conditions function as the appearance classical sets of assignments have in the dynamic setting. Resettings of variables are relations of the form $[\exists x]$, where

$$f[\exists x]g :\Leftrightarrow f \text{ and } g \text{ are the same on all variables except possibly } x.$$

The relations thus generated by tests and resettings are *DPL*-relations, i.e., relations typically definable in the language of *DPL*, a variant of predicate logic invented by Jeroen Groenendijk and Martin Stokhof (see [4]). Which *mDIL*-principles are valid for the restricted set of relations?

In this paper we will make this question precise and prove that the same sequents are valid in both cases.[1]

Here is a sketch of the argument. One first shows that if two structures are bisimilar, then they satisfy the same sequents. This is immediate by the well known fact that composition and dynamic implication are *safe* for bisimulation. Then one shows that for every structure $\mathfrak{N}$, there is a structure $\mathfrak{O}$ which is bisimilar to $\mathfrak{N}$ and in which every relation can be generated from conditions and resettings.

[1] This result provides a partial confirmation of a conjecture of Johan van Benthem and Giovanna Ceparello. See [2], conjecture 9.

The domain of $\mathfrak{O}$ is the cartesian product of the domain of $\mathfrak{N}$ with itself. A relation R of $\mathfrak{N}$ is mapped to a relation $\widehat{R}$ of $\mathfrak{O}$, where:

$$\langle a, b\rangle \widehat{R} \langle c, d\rangle :\Leftrightarrow aRc \text{ and } c = d.$$

These relations $\widehat{R}$ can be specified in the *DPL*-language by:

$$\exists y.R(x, y).\exists x.(x = y)$$

This formula means: reset y arbitrarily and then test whether it gives an appropriate output for R. If so, reset x to the value of y. The presence of formula shows that *DPL* has the resources to transform a *static* representation of a relation to a *dynamic* one. To do this we need the *auxiliary variable* y. We will see that this is unavoidable. It is typical for *DPL* that it does not have truly *local variables*, i.e., variables that can be used for temprary storage and then can be restored to their original content —unless we use some other variable to store this content, but then this variable ....

The work in the present paper illustrates the usefulness of notions like bisimulation and safety in thinking about relational languages. Moreover, the paper shows one way one could think about the use of *auxiliary variables in DPL*: they are things to be divided out modulo bisimulation.

2 The problem stated

We provide the necessary definitions to make a precise statement of our problem possible at all. We start with introducing some useful relational notions.

Definition 2.1 Let X be any non-empty set.

1. *Rel(X)* is the set of binary relations on X, i.e., $Rel(X) := \wp(X \times X)$
2. Let $R, S \in Rel(X)$. The composition $R \circ S$ of R and S is defined by: $x(R \circ S)y :\Leftrightarrow \exists z\ xRzSy$
3. Let $R, S \in Rel(X)$. The dynamic implication $(R{\rightarrow}S)$ between R and S is defined by:

   $$x(R{\rightarrow}S)y :\Leftrightarrow x = y \text{ and } \forall z(xRz \Rightarrow \exists u\ zSu).$$

   Our use of $\rightarrow$ here overloads the symbol, since we also use it for implication in the objectlanguage.
4. id_X is the identity relation. A relation R is a *condition* if $R \subseteq id_X$.

∎

The notion of dynamic implication was first introduced by Hans Kamp in his pioneering paper [6]. We turn to the definition of *DPL*.

Definition 2.2 1. A *DPL*-language $\mathcal{L}$ is a structure $\langle Pred, Ar, Var\rangle$, where *Pred* is a set of predicate symbols; *Ar* is a function from *Pred* to the natural numbers (including 0); *Var* is a, possibly empty, set of variables.

2. The set of $\mathcal{L}$*-formulas*, $For_{\mathcal{L}}$, is the smallest set such that:
   - $P(v_1, \cdots, v_n)$ is in $For_{\mathcal{L}}$, for $P \in Pred$ with $Ar(P) = n$ and $v_1, \ldots, v_n \in Var$
   - $\top$, $\bot$, $v = w$, $\exists v$ are in $For_{\mathcal{L}}$ for $v, w \in Var$
   - If $\phi, \psi \in For_{\mathcal{L}}$, then so are $\phi.\psi$ and $(\phi \rightarrow \psi)$

I feel that it is more faithful to the semantics to leave out the brackets in the formation rule for the dot *officially*, but nothing important hangs on this choice in this paper. We get an ambiguous syntax, but still unique meanings, since the operation of composition is associative. We use $\neg(\phi)$ and $\forall v(\phi)$ as abbreviations of, respectively, $(\phi \to \bot)$ and $(\exists v \to \phi)$. If x and y are distinct variables, we write $[x := y]$ for: $\exists x.x = y$. An alternative notation for $\exists v$, is $[v :=?]$ (random reset).

3. A *DPL*-model $\mathfrak{M}$ for a *DPL*-language $\mathcal{L}$ is a structure $\langle D, I\rangle$, where D is a non-empty set, the *domain* of $\mathfrak{M}$; I is a function which assigns to each predicate symbol P of $Pred_{\mathcal{L}}$ an $Ar(P)$-ary relation on D.

4. Let a *DPL*-model $\mathfrak{M}$ be given. $Ass_{\mathfrak{M}}$, the set of *assignments* for $\mathfrak{M}$, is D^{Var}.

5. Let a *DPL*-model $\mathfrak{M}$ for a *DPL*-language $\mathcal{L}$ be given. We define an interpretation function $[.]_{\mathfrak{M}} : For_{\mathcal{L}} \to Rel(Ass_{\mathfrak{M}})$ as follows.

   - $f[P(v_1, \cdots, v_n)]_{\mathfrak{M}} g :\Leftrightarrow f = g$ *and* $\langle f(v_1), \cdots, f(v_n)\rangle \in I(P)$
   - $f[\top]_{\mathfrak{M}} g :\Leftrightarrow f = g$
   - $f[\bot]_{\mathfrak{M}} g :\Leftrightarrow f \neq f$
   - $f[v = w]_{\mathfrak{M}} g :\Leftrightarrow f = g$ and $f(v) = f(w)$
   - $f[\exists v]_{\mathfrak{M}} g :\Leftrightarrow \forall w \in Var\backslash\{v\}\ f(w) = g(w)$
   - $[\phi.\psi]_{\mathfrak{M}} := [\phi]_{\mathfrak{M}} \circ [\psi]_{\mathfrak{M}}$
   - $[(\phi \to \psi)]_{\mathfrak{M}} := ([\phi]_{\mathfrak{M}} \to [\psi]_{\mathfrak{M}})$

6. A model $\mathfrak{M}$ satisfies a formula ϕ at an assignment f or $\mathfrak{M}, f \models \phi$ iff $\exists g \in Ass_{\mathfrak{M}}\, f[\phi]_{\mathfrak{M}} g$, i.o.w., if $[\phi]_{\mathfrak{M}}$ *accepts* f.

7. We define validity in a given *DPL*-model $\mathfrak{M}$ by: $\phi \models_{\mathfrak{M}} \psi :\Leftrightarrow \forall f, g(f[\phi]_{\mathfrak{M}} g \Rightarrow \exists h\ g[\psi]_{\mathfrak{M}} h)$ As usual, we define $\phi \models \psi$ if $\phi \models_{\mathfrak{M}} \psi$ for all models $\mathfrak{M}$ that are appropriate for the given language.

8. A binary relation R is *definable* in a *DPL*-model $\mathfrak{M}$ for a language $\mathcal{L}$ if there is an $\mathcal{L}$-formula ϕ, which defines R, i.e., $R = [\phi]_{\mathfrak{M}}$.

□

We will often suppress the subscript $\mathfrak{M}$, when the model is clear from the context. We could extend the *DPL*-language with function symbols by copying the way this is done in ordinary predicate logic. However, for the kind of result we are after such an extension is immaterial, since the usual trick to eliminate function symbols works also in *DPL* —with a small twist. E.g., $P(f(g(x)))$ will be translated to: $\neg\neg(\exists u.G(x,u).\exists v.F(u,v).P(v))$.

Example 2.3 [with Kees Vermeulen] We show that in no *DPL*-model with at least two elements in D we can define the relation $\neq$ as resetting relation. Suppose we could. Say, ϕ is the defining formula. Since $\neq$ is not a condition ϕ must contain at least one $\exists v$ (not confined in an implication). Let $\exists x$ be the first such. So, ϕ has the form $C.\exists x.\chi$, where C stands for a condition. Since every f can be succesfully continued via $\neq$, no f is weeded out in advance. Hence, C must be equivalent to $\top$. So we may write ϕ as $\exists x.\chi$. Consider any f and g with $f \neq g$ and $f[\exists x]g$. (Since our domain has at least two elements such f and g exist.) We have $f[\exists x.\chi]g$. But, then, we also have $g[\exists x.\chi]g$, contradicting the assumption that our formula defines $\neq$. (For a more extensive discussion of the question which relations are *DPL*-definable see [7].) □

A typical feature of *DPL* is that the predicate symbols do not take all possible *DPL*-meanings as possible values. As a consequence, we do not generally have: $\phi \models \psi \Rightarrow \phi[P := \chi] \models \psi[P := \chi]$, where P is a predicate symbol and where no free variables in χ are bound in ϕ, ψ. The simplest example is as follows. We *do* have $P \models P$, but *not* $Qx.\exists x.\neg Qx \models Qx.\exists x.\neg Qx$.

We now introduce relational models, where every possible relation can occur as atomic meaning.

Definition 2.4 We define the relational theory *minimal Dynamic Implication Logic*, *mDIL*, as follows.

1. An *mDIL*-language $\mathcal{K}$ is a structure $\langle Rels \rangle$, where $Rels$ is a, possibly empty, set of atomic relation symbols.

2. The set of *$\mathcal{K}$-formulas*, $For_{\mathcal{K}}$, is the smallest set such that:

   - R is in $For_{\mathcal{K}}$, for $R \in Rels$
   - $\bot$, δ are in $For_{\mathcal{K}}$
   - If $\phi, \psi \in For_{\mathcal{K}}$, then so are $\phi.\psi$ and $(\phi \rightarrow \psi)$

3. An *mDIL*-model $\mathfrak{N}$ for a *mDIL*-language $\mathcal{K}$ is a structure $\langle D, I \rangle$, where D is a non-empty set, the *domain* of $\mathfrak{M}$; I is a function which assigns to each atomic relation symbol R of $Rel_{\mathcal{K}}$ a binary relation on D

4. Let an *mDIL*-model $\mathfrak{N}$ for an *mDIL*-language $\mathcal{K}$ be given. We define an interpretation function $[.]_{\mathfrak{N}} : For_{\mathcal{K}} \rightarrow Rel(D)$ as follows: $[R] := I(R)$, $[\bot] := \emptyset$, $[\delta] := id_D$, $[\phi.\psi] := [\phi] \circ [\psi]$, $[(\phi \rightarrow \psi)] := ([\phi] \rightarrow [\psi])$

5. An *mDIL*-model $\mathfrak{N} = \langle D, I \rangle$ is *DPL*-definable (*Dd*), if there is a *DPL*-model $\mathfrak{M}$, such that D is (isomorphic to) $Ass_{\mathfrak{M}}$ and each *I(R)* is definable in $\mathfrak{M}$.

6. We define validity in *mDIL* by: $\phi \models_{\mathfrak{N}} \psi :\Leftrightarrow \forall f, g(f[\phi]_{\mathfrak{N}} g \Rightarrow \exists h\ g[\psi]_{\mathfrak{N}} h)$. $\phi \models \psi$ if $\phi \models_{\mathfrak{N}} \psi$ for all models $\mathfrak{N}$ for the appropriate language. $\phi \models_{Dd} \psi$ iff $\phi \models_{\mathfrak{N}} \psi$ for all *DPL*-definable models $\mathfrak{N}$, appropriate for the given language.

□

Let me briefly comment on a subtle notational difference between *DPL*-language and *mDIL*-language. The *DPL*-predicate symbols, including $\top$ and $\bot$ range over sets/conditions. Thus $\top$ stands for the *largest condition* in the subset ordering. When viewed as a binary relation, the denotation of $\top$ is simply the identity relation. In *mDIL* the atomic letters, stand for binary resetting relations. Thus here $\top$ would, reasonably, stand for the universal relation. Hence, we use δ for the identity relation in *mDIL*.

mDIL was studied by Patrick Blackburn and Yde Venema in their paper [3]. They provide a system of reasoning for *mDIL*. A *DPL*-definable *mDIL*-model $\mathfrak{N}$ is just an *mDIL*-model in which the atomic relations can be generated from conditions and resettings. Note that it follows that *all* relations of $\mathfrak{N}$ can be generated from conditions and resettings. Finally we have reached the stage, where we can officially state our problem:

Do we have $\models_{Dd} = \models$ for *mDIL*?

We end this section by describing an equivalent way of introducing $\models_{Dd}$. Let a *mDIL*-language $\mathcal{K}$ be given. A translation $(.)^*$ from $\mathcal{K}$ to a *DPL*-language $\mathcal{L}$ is a function from the formulas of $\mathcal{K}$ to the formulas of $\mathcal{L}$, such that $(.)^*$ commutes with $\bot$, . and $\rightarrow$ and sends δ to $\top$. We have for formulas ϕ, ψ in $\mathcal{K}$: $\phi \models \psi \Leftrightarrow$ for all $(.)^*$ (with domain $\mathcal{K}$) $\phi^* \models \psi^*$.

To solve our problem we will need some facts about transition systems and bisimulations. These are presented in the next section.

3 Transition Systems

To solve our problem we will need some facts about transition systems and bisimulations. These are presented in the present section.

Definition 3.1 1. A transition system $\mathfrak{G}$ is a structure $\langle S, A, \longrightarrow\rangle$. Here, S is a non-empty set, the set of *states*. A is a (possibly empty) set, the set of *labels*. $\longrightarrow$ is a function from A to the binary relations on S. We write $s \stackrel{a}{\longrightarrow} t$ for: $\langle s,t\rangle \in \longrightarrow(a)$. Note that our transition systems are "unrooted".

2. Consider two transition systems $\mathfrak{G}$ and $\mathfrak{H}$. Suppose $A_{\mathfrak{G}} = A_{\mathfrak{H}}$. A *bisimulation* $\mathcal{B}$ between $\mathfrak{G}$ and $\mathfrak{H}$ is a relation between $S_{\mathfrak{G}}$ and $S_{\mathfrak{H}}$, such that whenever $s\mathcal{B}s'$ and $s \stackrel{a}{\longrightarrow}_{\mathfrak{G}} t$, there is a t' with $t\mathcal{B}t'$ and $s' \stackrel{a}{\longrightarrow}_{\mathfrak{H}} t'$ (*the zig property*) and such that whenever $s\mathcal{B}s'$ and $s' \stackrel{a}{\longrightarrow}_{\mathfrak{H}} t'$, there is a t with $t\mathcal{B}t'$ and $s \stackrel{a}{\longrightarrow}_{\mathfrak{G}} t$ (*the zag property*). $\mathcal{B}$ is *total* if $dom(\mathcal{B}) = S_{\mathfrak{G}}$. $\mathcal{B}$ is *surjective* if $range(\mathcal{B}) = S_{\mathfrak{H}}$. $\mathcal{B}$ is *full* if $\mathcal{B}$ is total and surjective.

3. Let $\mathfrak{G}$ and $\mathfrak{H}$ have the same set of labels. Define:

   - $\mathcal{B} : \mathfrak{G} \preceq \mathfrak{H} :\Leftrightarrow \mathcal{B}$ is a total bisimulation between $\mathfrak{G}$ and $\mathfrak{H}$
   - $\mathfrak{G} \preceq \mathfrak{H} \Leftrightarrow$ for some $\mathcal{B}$ $\mathcal{B} : \mathfrak{G} \preceq \mathfrak{H}$
   - $\mathfrak{G} \equiv \mathfrak{H} \Leftrightarrow \mathfrak{G} \preceq \mathfrak{H}$ and $\mathfrak{H} \preceq \mathfrak{G}$

❑

It is easy to check that —for transition systems on a fixed set of labels— $\preceq$ is a partial pre-ordering. Bisimulations are closed under union and converse. So when $\mathfrak{G} \equiv \mathfrak{H}$, then there is a full bisimulation between $\mathfrak{G}$ and $\mathfrak{H}$.

Fix an *mDIL*-language $\mathcal{K}$. Consider an *mDIL* model $\mathfrak{N}$. We associate two transition systems $\mathfrak{G} := T_0(\mathfrak{N})$ and $\mathfrak{H} := T_1(\mathfrak{N})$ to $\mathfrak{N}$ by taking:

- $S_{\mathfrak{G}} := S_{\mathfrak{H}} := D_{\mathfrak{N}}$
- $A_{\mathfrak{G}} := Rel_{\mathcal{K}}$, $A_{\mathfrak{H}} := For_{\mathcal{K}}$
- $\longrightarrow_{\mathfrak{G}} := I_{\mathfrak{N}}$, $\longrightarrow_{\mathfrak{H}} := [.]_{\mathfrak{N}}$

We state some simple facts.

Theorem 3.2 *Consider two $\mathcal{K}$-models $\mathfrak{N}$ and $\mathfrak{O}$. Suppose $T_0(\mathfrak{N}) \preceq T_0(\mathfrak{O})$. Then, $T_1(\mathfrak{N}) \preceq T_1(\mathfrak{O})$.*

Proof

The verification is immediate. The basic insight is that 'adding the identity transition', 'adding the empty transition', composition and dynamic implication are *safe* for bisimulations. (The fact that, e.g., $\circ$ is safe, means that if $\mathcal{B}$ is a bisimulation for the transitions α and $beta$, then it also a bisimulation for $\alpha \circ beta$.) Safety is studied and characterized in [1] and [5]. ❑

We write $\mathfrak{N} \preceq \mathfrak{O}$ for: $T_0(\mathfrak{N}) \preceq T_0(\mathfrak{O})$.

Theorem 3.3 *Consider two $\mathcal{K}$-models $\mathfrak{N}$ and $\mathfrak{O}$. Suppose $\mathfrak{N} \preceq \mathfrak{O}$. Then, $\models_{\mathfrak{O}} \subseteq \models_{\mathfrak{N}}$.*

Proof

Let $\mathcal{B}$ witness $\mathfrak{N} \preceq \mathfrak{O}$. Note that we have: $\mathcal{B} : T_1(\mathfrak{N}) \preceq T_1(\mathfrak{O})$. Suppose $\phi \models_{\mathfrak{O}} \psi$ and $d[\phi]_{\mathfrak{N}} e$. Since $\mathcal{B}$ is a total bisimulation, we can find d',e' with: $d\mathcal{B}d'$, $e\mathcal{B}e'$ and $d'[\phi]_{\mathfrak{O}} e'$. Since $\phi \models_{\mathfrak{O}} \psi$, there is an f' with $e'[\psi]_{\mathfrak{O}} f'$. We have $e\mathcal{B}e'$ and $e'[\psi]_{\mathfrak{O}} f'$, so, by the zag property, there is an f with $f\mathcal{B}f'$ and $e[\psi]_{\mathfrak{N}} f$. ❑

4 The problem solved

We want to show that $\models_{Dd} = \models$. By theorem 3.3 it is sufficient to show that for every $\mathcal{K}$-model $\mathfrak{N}$ there is a *DPL*-definable model $\mathfrak{O}$ such that $\mathfrak{N} \equiv \mathfrak{O}$.

Consider an *mDIL*-model $\mathfrak{N}$ for $\mathcal{K}$. We write $R_{\mathfrak{N}}$ for $I_{\mathfrak{N}}(R)$, etcetera. We construct $\mathfrak{O}$ as follows.

- $D_{\mathfrak{O}} := D_{\mathfrak{N}} \times D_{\mathfrak{N}}$. We conveniently 'identify' $D_{\mathfrak{O}}$ with $D_{\mathfrak{N}}^{\{x,y\}}$, where x and y are distinct variables.
- $R_{\mathfrak{O}} := \widehat{R_{\mathfrak{N}}}$, where $\langle a,b\rangle \widehat{R_{\mathfrak{N}}} \langle c,d\rangle :\Leftrightarrow aR_{\mathfrak{N}}c$ and $c = d$

Let $a\mathcal{B}\langle b,c\rangle :\Leftrightarrow a = b$. Clearly $\mathcal{B}$ is a full bisimulation witnessing $\mathfrak{N} \equiv \mathfrak{O}$. Finally we show that $\mathfrak{O}$ is *DPL*-definable. Consider the *DPL*-language $\mathcal{L}$ and the $\mathcal{L}$-model $\mathfrak{M}$ given by:

- $Pred_{\mathcal{L}} := Rels_{\mathcal{K}}$
- $Ar_{\mathcal{L}}(R) := 2$
- $Var_{\mathcal{L}} = \{x, y\}$
- $D_{\mathfrak{M}} := D_{\mathfrak{N}}$
- $I_{\mathfrak{M}}(R) := I_{\mathfrak{N}}(R)$

Note the crucial subtlety that R stands for a *set of pairs* in $\mathfrak{M}$, but for a resetting relation in $\mathfrak{N}$. We can now define the resetting relation $R_{\mathfrak{O}}$ in $\mathfrak{M}$, by: $\exists y.R(x,y).\exists x.x = y$, or, using an abbreviation: $\exists y.R(x,y).[x := y]$. We leave the easy check that this formula does deliver the promised goods to the reader.

Our formula effects a *static-dynamic conversion*. It does this by employing an auxiliary variable y. Note that the original contents stored under y are irrevocably lost. We proceed to illustrate, that the use of an auxiliary variable cannot be avoided. We have considered the *DPL*-definable relations. This class can be refined in a natural way by counting the number of variables in the *DPL*-language we are considering. Thus we can talk about the *DPL*-1-definable relations (*1Dd*), the *DPL*-$\aleph_0$-definable relations ($\aleph_0 Dd$), etcetera. Clearly, we have shown that: $\models_{2Dd} = \models$. The result is is preserved if we look at numbers bigger than 2, since for any $n \geq 2$: $\models \subseteq \models_{nDd} \subseteq \models_{2Dd} \subseteq \models$. We do not have $\models_{1Dd} = \models$, as is shown by the following theorem.

Theorem 4.1 *We have* $\neg\neg(\phi.\phi) \models_{1Dd} \phi.\phi.\phi$*, but not* $\neg\neg(\phi.\phi) \models \phi.\phi.\phi$.

Proof

To prove the first half of the result, we work in a language with one variable x and in a model for this language. Suppose ψ stands for a condition. By a remark of Kees Vermeulen, we have: $[\exists x.\psi.\exists x] = [\neg\neg(\exists x.\psi).\exists x]$. It follows that ϕ can —salva significatione— be rewritten as $C.\exists x.C'$, where C and C' stand for conditions. Now it is easy to see that both $\phi.\phi$ and $\phi.\phi.\phi$ are equivalent to $\phi.\neg\neg(\exists x.C'.C)$. ¿From this insight, the first half of our result is immediate. The second half is proved by considering countermodel: $\bullet \xrightarrow{\phi} \bullet \xrightarrow{\phi} \bullet$. ❑

Remark 4.2 We provide some remarks and formulate some questions.

1. Note that our result still works for any extension of *mDIL* with bisimulation safe operations. Thus we could extend our language with $\cup$ (union of relations, indeterministic choice). On the other hand $\cap$ is not safe. So what happens, when we add $\cap$? The operations that are safe for (partial) bisimulations were characterized by Johan van Benthem in [1]. We have a bit more here, since *full* bisimulation is the relevant notion. For example, 'adding the universal relation' is safe for full bismimulation, but not for (partial) bisimulation.[2] So we could add $\top$, where $\top$ stands for the universal relation, to the *mDIL*-language, preserving our result.

2. If we replace $\models$ in our question by $\subseteq$ (the subset relation between binary relations), the above argument does not work any more, since the analogue of theorem 3.3 fails. What happens in this case?

3. An alternative route would have been to take: $\langle a, b\rangle \widehat{R_{\mathfrak{N}}} \langle c, d\rangle :\Leftrightarrow aR_{\mathfrak{N}}c$ and define this relation by: $\exists y.R(x, y).\exists x.x = y.\exists y$.

4. Our result illustrates that *modulo bisimulation* is a good way of thinking about *auxiliary variables* in *DPL*.

□

References

[1] J.F.A.K. van Benthem. Program constructions that are safe for bisimulation. CSLI report 93-179, CSLI, Stanford University, Stanford, July 1993. to appear in Proceedings Logic Colloquium. Clermont Ferrant 1994, North Holland, Amsterdam.

[2] J.F.A.K. van Benthem and G. Ceparello. Tarskian variations. dynamic parameters in classical semantics. CS R9419, CWI, Amsterdam, March 1994.

[3] P. Blackburn and Y. Venema. Dynamic squares. Logic Group Preprint Series 92, Department of Philosophy, Utrecht University, Heidelberglaan 8, 3584 CS Utrecht, June 1993. to appear in Journal of Philosophical Logic.

[4] J. Groenendijk and M. Stokhof. Dynamic predicate logic. *Linguistics and Philosophy*, 14:39–100, 1991.

[5] M. Hollenberg. General safety for bisimulation. unpublished manuscript, 1995.

[6] H. Kamp. A theory of truth and semantic representation. In J. Groenendijk et al., editors, *Formal Methods in the Study of Language*, Amsterdam, 1981. Mathematisch Centrum.

[7] A. Visser. Contexts for dynamic predicate logic. Logic Group Preprint Series 143, Department of Philosophy, Utrecht University, Heidelberglaan 8, 3584 CS Utrecht, October 1995.

[2] Marco Hollenberg has proved that the operations safe for full bisimulation are obtained by adding the universal relation to the basic repertoire in van Benthems result. (See [5].)