Preferential Semantics for Action Specifications in First-order Modal Action Logic[1]

Jan Broersen[2]
Faculty of Mathematics and Computer Science, *Vrije Universiteit*
De Boelelaan 1081a, 1081 HV Amsterdam, the Netherlands
broersen@cs.vu.nl

Roel Wieringa
Faculty of Computer Science, *University of Twente*
P.O. Box 217, 7500 AE Enschede, the Netherlands
roelw@cs.utwente.nl

Abstract

In this paper we investigate preferential semantics for declarative specifications in a First Order Modal Action Logic. We address some well known problems: the frame problem, the qualification problem and the ramification problem. We incorporate the assumptions that are inherent to both the frame and qualification problem into the semantics of the modal Action Logic by defining orderings over Dynamic Logic models. These orderings allow us to identify for each declarative Dynamic Logic action specification a unique intended model.

1 Introduction

Golshani et al. [10] introduced a version of first-order dynamic logic called *Modal Action Logic* (MAL) to specify database updates declaratively. MAL was used later to specify requirements on the external behavior of software and to specify descriptive and prescriptive (deontic) constraints on the external behavior systems in general [14, 13, 12, 20]. Meyer and Wieringa [21, 23, 22] studied a closely related version of *Dynamic Logic* (DL), which was also used to specify descriptive and deontic constraints on database updates and object behavior declaratively. Although these papers show the wide applicability of MAL in the specification of agent behavior, they do not show how we can reason about actions of agents and their effects. In this paper we take a first step towards performing this type of reasoning by defining intended minimal models for a certain class of agent specifications. We argue that these models allow for a form of preferential model checking, which we intend to study in a sequel to this work.

We focus on a specific subset of MAL-formulas to specify the most relevant aspects of (nondeterministic!) actions e.g. their effects, guards on their occurrence and global static invariants. We do not specify sufficient conditions on occurrence of actions. We argue that the role they play can better be handled by a suitable definition of intended (prefered) model in which actions can take place whenever this is compatible with the global invariants and effects. A second problem we address is the problem of minimal change. We define a preference relation over MAL-models that selects the models in which the effects are interpreted under a minimal change condition. We investigate the interplay between both preference relations and define an intended model that is prefered with respect to both orderings. This gives us an exact unique meaning (model) for each specification. The intended model provides the basis for reasoning about properties of the specified actions, by means of model checking.

In section 2, we define our variant of MAL and in section 3, we consider agents subject to a number of *static constraints*, which may be descriptive or prescriptive, and actions that are subject to a number of declarative constraints on their *effects* and declarative *guards* on their occurrence. Corresponding to these two constraints on actions, we define in section 3 two preference relations on models, which order models on minimal change and maximal reachability, respectively. We study properties of these preference relations for three classes of action specifications. In section 4 we compare our results with other work and in section 5, we discuss how our work prepares the way for preferential model checking. This paper extends earlier work done on action specification in *Propositional Dynamic Logic*. In this workshop paper, all proofs are omitted. Most of them are given in the internal report [3] from which this paper is derived.

[1]Partly supported by Esprit Working Group Aspire, contact nr. 22704.
[2]Supported by USF-VU as part of the SINS contract.

2 Modal Action Logic

The syntactic elements of the Modal Action Language we use are the **punctuation symbols:** { ')', '(', ',' }, a set of **variable symbols** $\mathcal{V}$, a set of **predicate symbols** $\mathcal{P}$ including one distinguished predicate denoted by $=$, a set $\mathcal{F}$ of **function symbols**, a set of **propositional connective symbols** $\{\neg, \vee\}$, the **quantification symbol** $\{\exists\}$, a set $\mathcal{A}$ of **atomic action symbols** and the **operation symbol** $\{\langle . \rangle\}$.

The intended use of the logic is the writing of action specifications for agents. A specific specification uses only a finite subset of the language. A **signature** Σ is thus defined as a specific combination of finite subsets of the predicate, function and atomic action symbols: $\Sigma = (P, F, A)$. Apart from a signature, a specifier will also have to provide arities for predicate and function symbols. The arity of a predicate or function prescribes a length for the term lists concatenated to it. A signature contains the symbols that are to be given an interpretation relative to a specification. All other symbols are given logical interpretations.

The next BNF-sentences (ad hoc extended to incorporate indexing of syntactic elements) define atomic and well formed formulas.

$$
\begin{array}{lll}
term & ::= & variable \mid constant \mid function^n \ (\ termlist_n\ ) \\
constant & ::= & function^0 \\
termlist_1 & ::= & term \\
termlist_n & ::= & termlist_{n-1}\ ,\ term \\
atom & ::= & predicate^n\ (\ termlist_n\ ) \\
\phi & ::= & atom \mid \top \mid \bot \mid \neg\ \phi\ \mid \phi\ \vee\ \psi \mid \forall\, variable\ \phi \mid \langle\ action\ \rangle\ \phi
\end{array}
$$

We use $\phi, \psi, \chi, \ldots$ as meta variables over well formed formulas. We apply the usual syntactic abbreviations. A **specification** $Spec = (\Sigma, \Phi)$ is a pair consisting of a signature Σ and a finite set Φ of well formed formulas over Σ.

To define the semantics of MAL we define the notion of MAL-structure.

Definition 1 *Given a signature $\Sigma = (P, F, A)$ a MAL-structure $\mathcal{S} = (S, \mathcal{I}_A, D, \mathcal{I}_P^S, \mathcal{I}_f^S)$ is defined as:*

- *S is a nonempty set of possible states*
- *$\mathcal{I}_A$ is a total function $A \rightarrow 2^{S \times S}$, that maps action symbols to relations over states.*
- *D is an arbitrary domain*
- *$\mathcal{I}_P^S$ is a function $S \rightarrow (P \rightarrow 2^{D^n})$, that for each state s maps predicate symbols to relations over the domain.*
- *$\mathcal{I}_f^S$ is a function $S \rightarrow (F \rightarrow (D^n \rightarrow D))$, that for each state s maps function symbols to functions on the domain.*

Combinations of states S and a relation $\mathcal{I}_A$, that are by definition common to many structures, are called 'frames' ($\mathcal{F}$).

We do not want to interpret actions over states that differ in their interpretation of free variables; actions can only influence the interpretation predicate and function symbols. Therefore we require that the assignment of free variables to domain elements is common to all states. This implies that we need a Domain that is common to all states. This is completely different from the situation in Dynamic Logic [11], where actions (programs) are interpreted to modify values (bindings) of free variables. In DL it is the variables that are interpreted different in different states, while the interpretation of predicates, functions and constants stays the same. Note however that function symbols of arity 0 can be used as variables in the sense of DL.

Definition 2 *Given a structure $\mathcal{S} = (S, \mathcal{I}_A, D, \mathcal{I}_P^S, \mathcal{I}_f^S)$ and a set of variables V, an assignment $\mathcal{I}_V$ is a function $V \rightarrow D$, assigning a domain element to each variable in V.*

The next definitions give the interpretation of well-formed MAL-formulas.

Definition 3 *Given a signature $\Sigma = (P, F, A)$, a structure $\mathcal{S} = (S, \mathcal{I}_A, D, \mathcal{I}_P^S, \mathcal{I}_f^S)$ and an assignment $\mathcal{I}_V$, an interpretation $\mathcal{I}_t^S$ of a term t in a state s is defined as:*
$\mathcal{I}_t^S = \mathcal{I}_V(t)$ in case t is a variable
$\mathcal{I}_t^S = \mathcal{I}_f^S(f^n)(\mathcal{I}_t^S(t_1), \ldots, \mathcal{I}_t^S(t_n))$ in case t has the form $f^n(t_1, \ldots, t_n)$

Definition 4 *Given a signature $\Sigma = (P, F, A)$, a structure $\mathcal{S} = (S, \mathcal{I}_A, D, \mathcal{I}_P^S, \mathcal{I}_f^S)$ and an assignment $\mathcal{I}_V$, validity of a wff ϕ in a state s of the structure is defined as:*

$\mathcal{S}, s, \mathcal{I}_V \models \bot$	*never*	
$\mathcal{S}, s, \mathcal{I}_V \models t_1 = t_2$	*iff*	$\mathcal{I}_t^S(s)(t_1)$ *returns the same domain element as* $\mathcal{I}_t^S(s)(t_2)$
$\mathcal{S}, s, \mathcal{I}_V \models P_i^n(t_1, \ldots, t_n)$	*iff*	$(\mathcal{I}_t^S(s)(t_1), \ldots, \mathcal{I}_t^S(s)(t_n)) \in \mathcal{I}_P^S(s)(P_i^n)$
$\mathcal{S}, s, \mathcal{I}_V \models \phi \vee \psi$	*iff*	$\mathcal{S}, s, \mathcal{I}_V \models \phi$ *or* $\mathcal{S}, s, \mathcal{I}_V \models \psi$
$\mathcal{S}, s, \mathcal{I}_V \models \neg\phi$	*iff*	*not* $\mathcal{S}, s, \mathcal{I}_V \models \phi$
$\mathcal{S}, s, \mathcal{I}_V \models \exists x\, \phi(x)$	*iff*	*for some* $d \in D$ *holds* $\mathcal{S}, s, \mathcal{I}_V\{x \mapsto d\} \models \phi(x)$
$\mathcal{S}, s, \mathcal{I}_V \models \langle a \rangle \phi$	*iff*	*for some* $s' \in S$ *holds* $(s, s') \in \mathcal{I}_A(a)$ *and* $\mathcal{S}, s', \mathcal{I}_V \models \phi$
$\mathcal{S}, s, \mathcal{I}_V \models \top$	*always*	

The modal formulas $\langle a \rangle \phi$, where a is an action, mean that there is a possible occurrence of a after which ϕ holds. A formula is S-valid if it is valid in all states of a structure S. In that case, we say the structure satisfies the formula and that the structure is a model of the formula. A formula is valid if it is S-valid for every structure S.

3 Action specification in MAL

We investigate the use of MAL for writing action specifications. The first observation we make is that we do not need formulas with nested modalities for the specification of atomic actions. Nested modalities correspond to properties of sequences of actions. We assume here that all relevant system properties can be specified by focusing on individual atomic actions. We distinguish four types of MAL-formulas:

- **Conditional postcondition formulas:** $\phi \rightarrow [a]\psi$ We say that ϕ is a sufficient precondition of a with respect to the postcondition ψ. We denote sets of these formulas by Φ_{post}.
- **Guard formulas:** $\langle a \rangle \top \rightarrow \chi$ We call χ a guard of a, equivalently, a necessary precondition for the possible occurrence of a. We denote sets of these formulas by Φ_{guard}.
- **Sufficient precondition formulas:** $\sigma \rightarrow \langle a \rangle \top$ We call σ a sufficient precondition of a. We denote sets of these formulas by Φ_{suff}.
- **Static constraints:** θ These are non-modal assertions that must be obeyed under any circumstance. We denote sets of these formulas by Φ_{ic}.

The formulas ϕ, ψ, θ, σ and χ contain no modal constructs and are in disjunctive normal form.

Proposition 1 *Any MAL formula that concerns only one action symbol and contains no nested modalities, can be decomposed in a conjunction of formulas of the forms defined above.*

This shows that the above formulas are the basic ones for the specification of actions in MAL.

Proposition 2 *Each MAL-formula that is a conjunction of the above formulas is satisfiable if and only if it is satisfiable by a model with unique states (states with a unique interpretation of predicate and function symbols).*

This means that there is actually no need to distinguish states from interpretations of the predicate and function symbols. We do not have to consider different states with equal interpretations within one model or different interpretations of predicates and functions in the same state. Therefore, from now on we identify states with the interpretation of predicate and function symbols. This justifies that structures $\mathcal{S} = (S, \mathcal{I}_A, D, \mathcal{I}_P^S, \mathcal{I}_f^S)$ are from now on denoted as $\mathcal{S} = (D, S, \mathcal{I}_A)$. (Remark: this can be seen as a first step in the selection of an **intended model**; in this first step all models with equal interpretations of predicate and function symbols in different states are left out. This is actually an intuitive thing to do, because the states of the model represent states of the agent, and agent states have no duplicates either.)

Since we focus on the specification of individual atomic actions, the previous proposition should not come as a surprise. If we would have allowed the specification of properties of sequences of atomic actions, truth conditions in states influence truth conditions in more then only the immediately 'neighboring' states. This means that states with equal interpretations of predicate and function symbols can not safely be identified.

There is however one severe difficulty with specifications made with the above set of formulas: the specification can easily become inconsistent. This is one of two reasons why we suggest to drop the sufficient preconditions. The following proposition explains why.

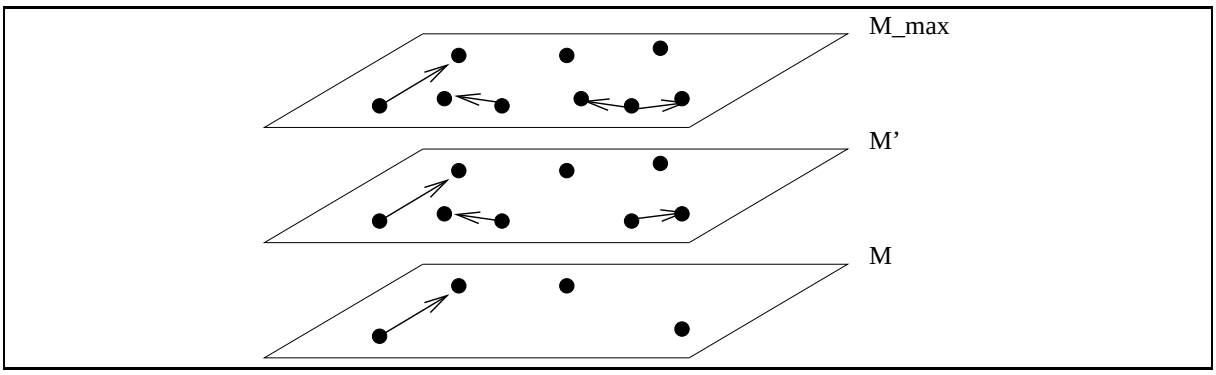


Figure 1: Comparing models on maximal reachability

Proposition 3 *Specifications that consist of conditional postcondition formulas Φ_{post}, guard formulas Φ_{guard} and static constraints Φ_{ic} are consistent if the static constraints are mutually consistent*

Also in LCM [6] the syntactic restriction that sufficient preconditions are absent is applied. Absence of this type of formulas implies transitions can always be dropped from models if they give rise to inconsistencies. Another way to deal with the problem is to weaken sufficient preconditions by specifying them as defaults [2] [17], which may be overridden.

In the next section we mention the second reason for dropping sufficient preconditions and define how the absence of sufficient preconditions formulas can be compensated for by a suitable notion of maximum reachability.

3.1 The qualification problem in MAL specifications

A common theme in the AI-literature is that it is not possible to foresee all necessary preconditions for the success of an action [8]. This problem is known as **the qualification problem**. This means that a specifier actually is never able to give a sufficient precondition for an action, which gives us a second reason to drop them from the set of specification formulas. Nevertheless we somehow need to compensate for the absence of sufficient preconditions. This is done by incorporating in the semantics the assumption that actions occur unless this contradicts guards (or static constraints, or conflicting postcondition axioms). We accomplish this by formalizing the qualification assumption in the notion of maximal reachability.

Definition 5 *Given a signature $\Sigma = (P, F, AA)$ and two structures $\mathcal{S}' = (D, S', \mathcal{I}'_{AA})$ and $\mathcal{S} = (D, S, \mathcal{I}_{AA})$*

$$
\begin{aligned}
&\mathcal{S}' \sqsubseteq_{mr} \mathcal{S} \textit{ iff} \\
&\quad S' \subseteq S \\
&\quad \textit{and} \\
&\quad \textit{for all } a \in AA,\ \mathcal{I}'_{AA}(a) \subseteq \mathcal{I}_{AA}(a)
\end{aligned}
$$

$\sqsubseteq_{mr}$ is a partial order on structures, because $\sqsubseteq_{mr}$ can easily seen to be transitive, reflexive and anti-symmetric. The ordering just 'prefers' as much states and transitions over them as possible, thus implementing the notion of 'maximal reachability'. In Figure 2 this is reflected by the fact that all transitions and states in lower models are also in the model at the top.

Proposition 4 *Let Φ be a set of formulas for which there is a model. Then there is a $\sqsubseteq_{mr}$-maximal model.*

Preferring $\sqsubseteq_{mr}$-maximal structures leads to non-monotonic entailment. An example of this is provided by the specification $\Phi = \{[a]A(p)\}$. Under interpretation over $\sqsubseteq_{mr}$-maximal structures, $\langle a \rangle A(p)$ is entailed by Φ. However, this is no longer true for $\Phi \cup [a]\neg A(p)$.

In the next section we formalize the notion of minimal change by defining a second ordering over labeled Kripke structures.

3.2 The frame problem in MAL specifications

Postcondition formulas describe the effect of an action. However, defining the effect of an action by means of a postcondition always causes **the frame problem** [4]. This problem states that when specifying a postcondition we only want to specify conditions that change. We do not want, and often are not able, to specify all the conditions that do not change as the result of an action. So we want to make the frame assumption that everything that has not been specified to change will not change.

In the following we will define an ordering over MAL-structures that formalizes this assumption. We want the ordering to compare models on the property of 'access to "closest" possible states'. We first define a notion of distance between states.

Definition 6 *Given a structure* $\mathcal{S} = (D, S, \mathcal{I}_A)$ *and two states* $s = (\mathcal{I}_P^S, \mathcal{I}_f^S)$ *and* $s' = (\mathcal{I}'_P, \mathcal{I}'_f)$ *in* S. *The difference* $Diff(s, s')$ *between them is defined as the set of predicate instances and function values in which the states differ* ($N(V)$ *is the cardinality of a set* V*):*

$$
\begin{aligned}
&Diff(s, s') \equiv \\
&\quad \{\langle P_i^n, (d_1, \ldots, d_n)\rangle \mid (d_1, \ldots, d_n) \in \mathcal{I}_P^S(P_i^n) \text{ and } (d_1, \ldots, d_n) \notin \mathcal{I}'_P(P_i^n)\} \cup \\
&\quad \{\langle P_i^n, (d_1, \ldots, d_n)\rangle \mid (d_1, \ldots, d_n) \notin \mathcal{I}_P^S(P_i^n) \text{ and } (d_1, \ldots, d_n) \in \mathcal{I}'_P(P_i^n)\} \cup \\
&\quad \{\langle f_i^n, (d_1, \ldots, d_n)\rangle \mid \mathcal{I}_f^S(f_i^n)(d_1, \ldots, d_n) \neq \mathcal{I}'_f(f_i^n)(d_1, \ldots, d_n)\}
\end{aligned}
$$

We want to use this measure of distance between states in the comparison between different MAL-structures. We define an ordering over Kripke structures that is connected to the definition of difference between states in such a way that structures where actions lead to 'closer' states are lower in the ordering. We will first give the two slightly different orderings, before we explain what they are really about.

Definition 7 *Given a signature* $\Sigma = (P, F, A)$ *and two structures* $\mathcal{S} = (D, S, \mathcal{I}_A)$ *and* $\mathcal{S}' = (D, S', \mathcal{I}'_A)$,

$$
\begin{aligned}
&\mathcal{S}' \sqsubseteq_{mc} \mathcal{S} \text{ iff} \\
&\quad S' = S \\
&\quad \text{and} \\
&\quad \forall a \in A,\ \forall s \in S,\ (\exists s' \in S,\ (s, s') \in \mathcal{I}'_A(a) \Leftrightarrow \exists s'' \in S,\ (s, s'') \in \mathcal{I}_A(a)) \\
&\quad \text{and} \\
&\quad \forall a \in A,\ \forall s \in S,\ \forall s' \in S,\ ((s, s') \in \mathcal{I}'_A(a) \Rightarrow \exists s'' \in S,\ ((s, s'') \in \mathcal{I}_A(a) \\
&\qquad\qquad \wedge N(Diff(s, s')) \leq N(Diff(s, s'')))
\end{aligned}
$$

$\sqsubseteq_{mc}$ is a pre-order on structures, because $\sqsubseteq_{mc}$ can easily seen to be transitive and reflexive. MC stands for minimal cardinality. A structure is called an **MC-model** of Φ if it is a $\sqsubseteq_{mc}$-minimal model of Φ. MC-models determine the minimal cardinality interpretation (semantics) of a specification (Σ, Φ).

Definition 8 *Given a signature* $\Sigma = (P, F, A)$ *and two structures* $\mathcal{S} = (D, S, \mathcal{I}_A)$ *and* $\mathcal{S}' = (D, S', \mathcal{I}'_A)$,

$$
\begin{aligned}
&\mathcal{S}' \sqsubseteq_{ms} \mathcal{S} \text{ iff} \\
&\quad S' = S \\
&\quad \text{and} \\
&\quad \forall a \in A,\ \forall s \in S,\ (\exists s' \in S,\ (s, s') \in \mathcal{I}_A(a) \Rightarrow \exists s'' \in S,\ (s, s'') \in \mathcal{I}'_A(a) \\
&\qquad\qquad \wedge (Diff(s, s') \subseteq Diff(s, s'') \vee Diff(s, s') \supseteq Diff(s, s'')))) \\
&\quad \text{and} \\
&\quad \forall a \in A,\ \forall s \in S,\ \forall s' \in S,\ ((s, s') \in \mathcal{I}'_A(a) \Rightarrow \exists s'' \in S,\ ((s, s'') \in \mathcal{I}_A(a) \\
&\qquad\qquad \wedge Diff(s, s') \subseteq Diff(s, s'')))
\end{aligned}
$$

$\sqsubseteq_{ms}$ is a pre-order on structures, because $\sqsubseteq_{ms}$ can easily seen to be transitive and reflexive. MS stands for minimal subset. A structure is called an **MS-model** of Φ if it is a $\sqsubseteq_{ms}$-minimal model of Φ. MS-models determine the minimal subset interpretation (semantics) of a specification (Σ, Φ).

The first requirement in both the $\sqsubseteq_{mc}$ and $\sqsubseteq_{ms}$ ordering is that structures can only be compared if they are based on the same set of states. In figure 1, where three models are compared on minimal change, this is reflected by the fact that all models contain the same set of black dots.

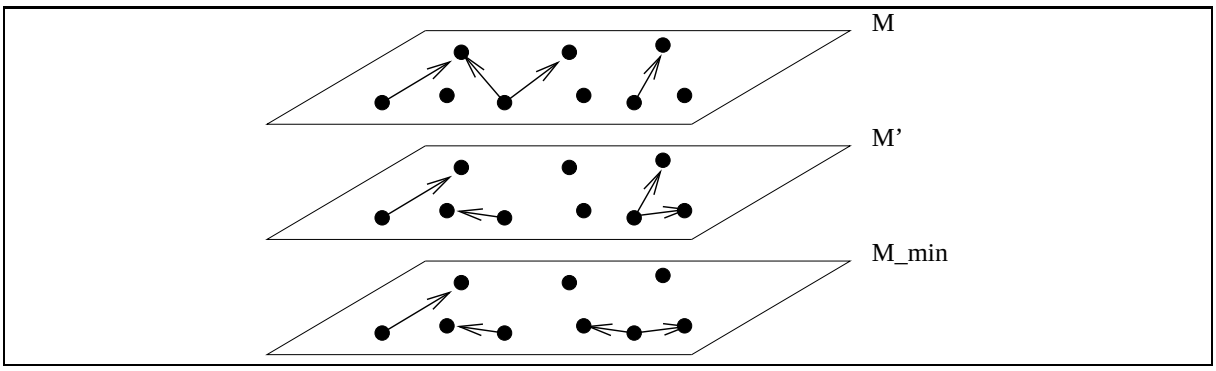


Figure 2: Comparing models on minimal change

The second requirement in both orderings forces that structures can only be compared if each transition from a state in one of the models actually corresponds to a transition from the same state in the other model, that is comparable under the minimal cardinality respectively the minimal subset criterion. For two transitions to be comparable under the minimal cardinality criterion, it is sufficient to demand that they leave from the same state; the comparison is just made on the *number* of changes that both transitions bring about. For two transitions to be comparable under the minimal subset criterion, the changes of the first must be a subset of the changes of the second or the other way around. In figure 1 this is represented by the fact that if an arrow leaves from some state in model there is also an arrow from this states in other models. (Note that the transitions may lead to different states. This is actually where we want to compare structures on; we want to prefer structures where transitions lead to "closer" states.) This is an intuitive criterion because we don't want this ordering to deal with the possible occurrence of transitions (actions); this ordering should compare structures purely on the 'length' of transitions. (This observation is also made by Brass and Lipeck [2] [17].)

The last requirement in the definition deals with this 'length' of transitions. In words it says: if $\mathcal{S}' \sqsubseteq_{mc} \mathcal{S}$ then for all transitions (s, s') in $\mathcal{S}'$ there is a transition (s, s'') in $\mathcal{S}$ that is 'longer'. In yet other words: if $\mathcal{S}' \sqsubseteq_{mc} \mathcal{S}$ then for corresponding transitions in corresponding states in the compared structures, the 'longest' transition in $\mathcal{S}'$ is always less or equal to the 'longest' transition in $\mathcal{S}$. In figure 1 the distance between dots represents the difference between states. The reason why this looks rather complicated is that we allow non-deterministic actions; we have to compare structures in which actions from one state can lead to several other states.

The difference between the semantics generated by both orderings will become more clear when we look at an example in section 3.5 and in the following section where we study minimal models for several specification classes. But we already can say that the minimal subset semantics is weaker than the minimal cardinality semantics, as is expressed by the following proposition.

Proposition 5 *An MC-model of Φ is also an MS-model of Φ.*

3.2.1 Minimal models for different specification classes

We will now investigate properties of minimal models for several classes of specification formulas. The properties we study are existence of minimal models, equivalence of the minimal cardinality and minimal subset criteria and determinism of minimal models.

We start of with the most general class we consider.

Definition 9 *We will refer to formulas Φ_{post}, Φ_{guard}, Φ_{ic} as defined before, as formulas of CLASS I*

An important property of minimal models for this most general class we consider is that if there is a model with a non-empty accessibility relation, then there is a *minimal* model with a non-empty accessibility relation.

Proposition 6 *Let Φ be a set of formulas of CLASS I that has a model $M = (D, S, \mathcal{I}_A)$ for which $\mathcal{I}_A(a) \neq \emptyset$ for all a in the formulas. Then there is an MS-model $M_{MS} = (D, S, \mathcal{I}'_A)$ of Φ for which $\mathcal{I}'_A(a) \neq \emptyset$ for all a in the formulas and an MC-model $M_{MC} = (D, S, \mathcal{I}''_A)$ of Φ for which $\mathcal{I}''_A(a) \neq \emptyset$ for all a in the formulas*

For this type of specifications MS-models and MC-models do not coincide and are not deterministic, as can be shown by a simple propositional example.

Example 1 *Consider the specification:*

$$[dial\_number](get\_connection \vee get\_busy\_tone))$$
$$get\_connection \rightarrow costs\_money$$
$$\langle dial\_number\rangle\top \rightarrow \neg costs\_money \wedge \neg get\_connection \wedge \neg get\_busy\_tone$$

Under the minimal subset criterion both the state where get(connection) holds and the state where get_busy_tone holds is reachable. Under the minimal cardinality criterion only the state where get_busy_tone holds is reachable. (Note that the constraint get_connection → costs_money is interpreted as being capable of inferring derived updates. We will come back to this in section 3.4)

We now turn our attention to quantification. Existential quantification is a source of non-determinism, as will be clear from the second class we study.

Definition 10 *the description of CLASS II*
formulas:

$$\begin{array}{ll}\Phi_{post}: & \phi \rightarrow [a]L_1 \wedge L_2 \wedge \ldots \wedge L_k \\ \Phi_{guard}: & \langle a\rangle\top \rightarrow \chi \\ \Phi_{ic}: & (M_1 \wedge M_2 \wedge \ldots \wedge M_l) \vee (N_1 \wedge N_2 \wedge \ldots \wedge N_m)\end{array}$$

with the L_h, M_i and N_j positive or negated atomic formulas (literals), and no restrictions on the quantification of variables whatsoever.

CLASS II is a subset of CLASS I. The difference is that postcondition formulas are made determinate (contain no disjunctive information) and constraints are limited to contain at maximum one disjunction in the disjunctive normal form. The following theorem holds for specifications of this type:

Proposition 7 *For formulas of CLASS II the minimal subset and minimal cardinality semantics coincide.*

Although the postcondition in the postcondition formulas of specifications of CLASS II are completely determinate, minimal models for these specifications are not deterministic. We will first define deterministic models and then give an example why minimal models for this class are not deterministic.

Definition 11 *A structure $\mathcal{S} = (D, S, \mathcal{I}_A)$ is **deterministic** if for each state $s \in S$ and for each a there is maximally one state s' such that $(s, s') \in \mathcal{I}_A(a)$.*

Example 2 *Consider the specification:*

$$[Shoot\_a\_gun]\exists something,\ Hit(something)$$

Obviously if the domain for the Predicate Hit has several "objects", there are many possible transitions possible from a certain state. More of these transitions can be present in minimal models (the minimality criteria do not minimize nondeterminism!) This reflects the fact that there are many things that are possibly hit. In the following we will see that the existential quantification is the only source for non-determinism in this specification.

The last class we consider precisely describes the class of formulas for which minimal models are deterministic. This is an important class. Since the minimal (intended) models are deterministic for these formulas, the only intention a specifier can have when providing formulas of this form, is to specify a deterministic system. Interpreting the formulas under a non-minimal semantics would allow for non-deterministic interpretation of the formulas, which is not what is intended.

Definition 12 *the description of CLASS III*
formulas:

$$\begin{array}{ll}\Phi_{post}: & \phi \rightarrow [a]L_1 \wedge L_2 \wedge \ldots \wedge L_k \\ \Phi_{guard}: & \langle a\rangle\top \rightarrow \chi \\ \Phi_{ic}: & (M_1 \wedge M_2 \wedge \ldots \wedge M_l) \vee (N_1 \wedge N_2 \wedge \ldots \wedge N_m)\end{array}$$

with the L_h, M_i and N_j positive or negated atomic formulas (literals), and with the restriction on the quantification that variables in L_h, M_i and N_j are all universally quantified.

This class is a subset of both CLASS I and CLASS II. The difference with CLASS II is that we allow only universal quantification. We can prove the property that for formulas of CLASS III, both MS-models and MC-models are deterministic.

Proposition 8 *For a specification Spec build with formulas from CLASS III, MS-models and MC-models are deterministic.*

The former property does not hold if we take as constraints the more general class of Horn-clauses, as is shown by the next example.

Example 3

$$C(j) \rightarrow [a]A(j)$$
$$\langle a\rangle\top \rightarrow \neg A(j) \wedge \neg B(j) \wedge C(j)$$
$$A(j) \wedge C(j) \rightarrow B(j)$$

The state $\{\overline{A}(j), \overline{B}(j), C(j)\}$ *has two possible follow-up states* $\{A(j), B(j), C(j)\}$ *and* $\{A(j), \overline{B}(j), \overline{C}(j)\}$ *We could of course add a distinction between base and other predicates to work around this.*

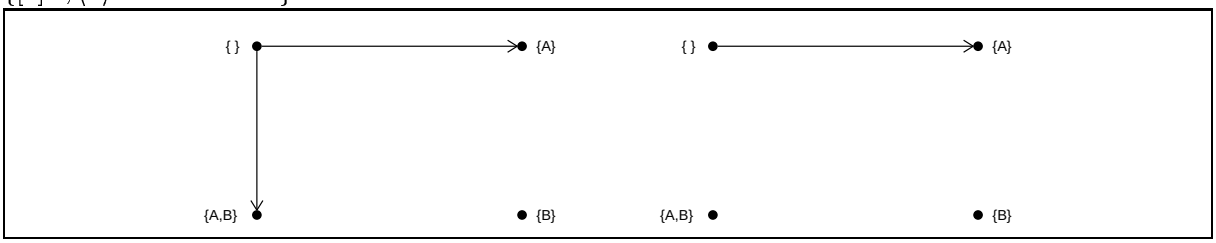
3.3 Min-Max-models

For the interpretation of specifications under both frame and qualification assumption we have to combine orderings. We do this by applying them one after the other. What, if any, is the "correct" (intuitive) order in which to do this? The answer can be found by looking at what the orderings are supposed to represent. The minimal change orderings concern the effect of actions independent from whether they occur or not. Maximal reachability deals with possible occurrence. This suggests that it is natural to apply minimal change first. First the minimal change ordering determines what actions we actually mean, by 'determining' their effects. And only after that we can 'talk' about the possible occurrence of actions. This motivates the following definition.

Definition 13 *Given a specification* (Σ, Φ)*, a* ***Min-Max-model*** *is defined as a* $\sqsubseteq_{mr}$*-maximal element of the set of MS/MC-models of* Φ*.*

The next (propositional) example shows that defining this in reverse order leads to a not-intended model.

We take a signature with $\mathcal{P} = \{A, B\}$, and $\mathcal{A} = \{a\}$, and the following set of formulas: $\Phi = \{[a]A, \langle a\rangle\top \rightarrow \neg A \wedge \neg B\}$.



The left picture shows the $\sqsubseteq_{mr}$-maximal structure that satisfies Φ. This structure is however not MS-satisfying. This shows it is not useful to apply maximality and minimality in this order. If we start by applying minimal change, we are left with two MS-models of the example formulas, the structure with no access to other states at all and the one shown in the right picture. Of these two clearly the one in the picture is $\sqsubseteq_{mr}$-maximal and the intended one.

The MS-models of a set of formulas Φ (we mean general formulas here) form a partially ordered set under the $\sqsubseteq_{mr}$-ordering. This set is not a lattice, as is shown by the following example.

Example 4 *Take the formula* $\neg(\langle a\rangle A(c) \wedge \langle a\rangle B(c))$ *and the two MS-satisfying structures* $\mathcal{S}$ *with* $\mathcal{I}_A$ *is* $\{\langle a, (\{\}, \{\langle A, \{d\}\rangle\})\rangle\}$ *and* $\mathcal{S}'$ *with* $\mathcal{I}'_A$ *is* $\{\langle a, (\{\}, \{\langle B, \{d\}\rangle\})\rangle\}$*. There is no MS-satisfying structure that is an upper bound for both structures* $\mathcal{S}$ *and* $\mathcal{S}'$*. A structure that might be seen as a candidate is* $\mathcal{S}''$ *with* $\mathcal{I}''_A$ *is* $\{\langle a, (\{\}, \{\langle B, \{d\}\rangle\})\rangle, \langle a, (\{\}, \{\langle A, \{d\}\rangle\})\rangle\}$*, because it is an upper bound under the* $\sqsubseteq_{mr}$*-ordering. But this structure is not MS-satisfying (not even satisfying).*

So in general there can be more Min-Max-models of a set of formulas Φ. However, for specification formulas of the restricted form we defined, we can prove that the MS-models form a complete lattice, which leads to the following proposition.

Proposition 9 *Each specification Φ_{trans} has a* ***unique Min-Max-model*** *if it has a model.*

In Min-Max-models, the maximization in most cases causes severe non-determinism. To see this we once more look at the following example.

Example 5 *Consider the specification:*

$$[Shoot\_a\_gun]\exists something,\ Hit(something)$$

If the variable "something" ranges over an infinite domain of objects, the action Shoot_a_gun from a given state branches to an infinite number of other states.

3.4 The interpretation of constraints

We will now shortly discuss two ways to interpret constraints. The first one is already applied in the earlier examples. There Constraints have the same "status" as other specification formulas, in the sense that in the process of selecting the intended model of a specification, they are accounted for from the beginning. So, we start with all models of all specification formulas, including the constraints, apply the minimal change criterion, and then the maximal reachability criterion. This way of interpreting constraints we call the "ramification semantics" of constraints, because it is an interpretation that leads to derived effects.

There is also the possibility to postpone the role of constraints to the second step in the selection procedure of the intended model. This means we start with all models of specification formulas minus the constraints, apply the minimal change criterion, then apply the criterion that models should satisfy the constraints, and finally apply the maximal reachability criterion.

By postponing the influence of constraints to the second step in this selection procedure, many transitions are deleted. This is because it may be the case that in the first step transitions are forced to reach closest possible worlds that in the second step may be found to violate a constraint. The transitions that are "deleted" are precisely the derived updates that are present in the ramification semantics. Therefore we call this interpretation of constraints simply the "constraint semantics".

3.5 Example

The following example specification is inspired by the Yale shooting problem:

$$\begin{array}{l} Gun\_loaded\_sharp \rightarrow [fire\_gun]Gun\_blown\_up \vee Bullet\_emitted \\ Gun\_loaded\_blank \rightarrow [fire\_gun]Gun\_blown\_up \vee Air\_and\_dust\_emitted \\ \rightarrow [fire\_gun]Big\_noise \\ \langle fire\_gun\rangle\top \rightarrow \neg Gun\_blown\_up \\ \langle fire\_gun\rangle\top \rightarrow Gun\_loaded\_sharp \vee Gun\_loaded\_blank \\ Bullet\_emitted \rightarrow Something\_is\_hit \\ \neg(Gun\_loaded\_sharp \wedge Gun\_loaded\_blank) \end{array}$$

First we conclude that the ramification semantics is the most intuitive interpretation for the constraints. Under the constraint semantics the condition $Something\_is\_hit$ can actually never change to true as a result of the action $fire\_gun$. It can only be true after the action $fire\_gun$ if it was already true in the state before $fire\_gun$ took place.

We will also have to choose between the the minimal subset and the minimal cardinality criterion. The choice is not too difficult if we look at the postcondition $Gun\_blown\_up \vee Bullet\_emitted$. Given the fact that we interpret constraints as possibly leading to ramifications, the truth of $Bullet\_emitted$ will imply the truth of $Something\_is\_hit$ in resulting states. This means that the action $fire\_gun$ usually 'has the choice' between making one atom true (Gun_blown_up) or two ($Bullet\_emitted$ and $Something\_is\_hit$). The minimal cardinality semantics will choose the first alternative which is really counterintuitive. The minimal subset semantics just makes no choice; both successor states are possible. This means that the minimal subset semantics is a more non-deterministic interpretation.

It is easy to give examples of how the action $fire\_gun$ can be qualified. Adding $\{Gun\_blown\_up\}$ or $\{\neg Big\_noise\}$ would leave us with no transitions at all. Adding $\{\neg Gun\_loaded\_sharp\}$ would result in transitions that never make $Something\_is\_hit$ true.

Finally we want to point out that the language to 'query' the intended model can be any language whose semantics can be defined using labeled Kripke structures, as is also argued by Lifschits e.a. [15]. As

an example of this we give some properties stated in Dynamic Logic using process operators like iteration test and sequence, that might be checked on the intended model of the example formulas.

$Gun\_loaded\_blank \rightarrow [fire\_gun^*]Gun\_loaded\_blank$

It says that if the gun was loaded with a blank it will stay loaded with blanks after a possibly infinite amount times of shooting.

$[Gun\_loaded\_sharp?\ ;\ fire\_gun](\neg Gun\_blown\_up \rightarrow Something\_is\_hit)$.

It says that if the gun is fired in a situation where it is loaded with a bullet, and as the result of it the gun will not blow up, then something is hit.

4 Comparison with other work

The frame problem and the associated problems of qualification and ramification are common themes in the AI literature on knowledge representation and reasoning about action and change [4].

Ryan [20] argues that structure consisting of a hierarchies of agents and sub-agents can be used to generate frame axioms automatically. The structure, that must be provided by the specifier, induces a notion of locality. This makes it possible to automatically generate two kinds of completion axioms: "local attribute axioms" stating that local attributes can only be affected by local actions and "local action axioms" stating that local actions can only affect local attributes. This last axiom involves a second order property, because it refers to attributes and not to their values. The proposed completion is however not complete. The first form of incompleteness is that within agents, all local attributes can still be affected by all local actions. The second form is that throughout the whole structure of agents all non-local attributes can still be affected by all non-local actions. Whether this gives rise to unintended results depends on the agent structure given by the specifier.

Reiter's approach to the frame problem [19] is to rewrite "effect axioms" to "successor state axioms". Effect axioms say for each action which predicates change their value if the action is performed. Successor state axioms say for each predicate which actions change when performed. Reiter's approach consists mainly of a change of focus from specific actions to specific predicates and a form of completion on them. An important difference with our approach is that Reiter uses sufficient conditions for the possibility of an action a, while we use only necessary conditions. So Reiter can actually never forbid actions from being possible in certain situations, he can only force them to be possible. As argued, we chose not to specify sufficient conditions, because this might cause problems in the presence of static constraints, which is exactly the problem Reiter et al. run into [16]. It is easy to show that Reiter's "successor state axioms", that are generated from "effect axioms" by performing a form of completion, can be expressed in MAL. However, Reiter's approach has several limitations. First of all it should be mentioned that in Reiter's approach the completion can only be performed on postcondition formulas that are determinate (no disjunctive postconditions). This means that this approach does not deal with non-deterministically specified effects. Second, the completion is not possible in the presence of static constraints that possibly contradict the effect. Third, the completion is not "complete". For details on this we refer to a full version of this paper [3].

Borgida et al. [1] take the perspective of the designer of specification languages and discuss ways to state that "nothing else changes" by syntactic as well as semantic means. Our work can be regarded as introducing a richer semantics for the specification language to capture this.

Giunchiglia, Kartha and Lifschitz introduce the action language $\mathcal{AR}$ [9], which is an extension of the language $\mathcal{A}$ [7], introduced by Gelfond and Lifschitz. $\mathcal{AR}$ differs from $\mathcal{A}$ in that it also deals with ramifications. There is a straightforward translation of most elements of the language $\mathcal{AR}$ into elements of (the propositional version of) our language. Under this translation their (minimal) interpretation function Res_D perfectly matches our Min-Max-models (because they only consider determinate postconditions there is no difference between the minimal subset or minimal cardinality criterion). The difference between their language and ours is that they can express that an effect possibly occurs (A **possibly changes** F **if** P) which can not be expressed in our language. On the other hand we can express that the effect of an action is a choice between two or more alterations ($[a](A \vee B)$), which can not be expressed in their language. Interesting is that both constructs are claimed to represent the non-deterministic aspect of actions.

Winslett's work on database update semantics [24] focuses on a model-oriented approach to updates, de-emphasizing the relation between the specification and the models. Instead, we base our semantics on the declarative semantics of a specification in MAL, which allows us to reason about updates in the same language.

Brass and Lipeck [2] [17] study action specification with the help of defaults. They also define orderings over modal interpretations. Frame and other assumptions are represented by formulas interpreted as defaults. This still puts the responsibility on the specifier to provide such formulas, which is not always desirable. Furthermore their models represent "action traces" and do not allow for non-determinism.

5 Conclusions and future work

In this paper we defined a preferential semantics for an important class of first order MAL formulas, that reflects the principles of minimal change and maximal reachability. Several results concerning more refined classes of specification formulas were reported.

We plan to compare existing completion procedures, such as the one suggested by Reiter, with our semantics, and to compare existing procedures for scenario generation and reachability analysis with our semantics. We also plan to investigate ways to generate the intended model (Min-Max-model) from a given specification. For that we will have to limit ourselves to finite domains and a finite number of actions. We will have to find a suitable representation for models and an algorithm that connects this representation to specifications. This opens possibilities for code generation and preferential model checking (to be read as "checking preferential models") using a model-checker like SMV [5] [18]. An example of the kind of properties that could be checked by preferential model checking was given at the end of the example section.

Another interesting future research area is the study of the notions of minimal change and maximal reachability in the context of concurrent actions and processes.

References

[1] A. Borgida, J. Mylopoulos, and R. Reiter. On the frame problem in procedure specifications. *IEEE Transactions on Software Engineering*, 21:785–798, 1995.

[2] S. Brass, U. W. Lipeck, and P. Resende. Specification of object behaviour with defaults. In G. Koschorreck U. W. Lipeck, editor, *Proceedings of the International Workshop on Information Systems – Correctness and Reusability (IS-CORE'93)*, pages 155–177. Informatik-Berichte 01/93, Universit t Hannover, Januari 1993.

[3] J.M. Broersen and R.J. Wieringa. Minimal semantics for action specifications in first-order dynamic logic. Technical Report IR-439, Faculty of Mathematics and Computer Science, Vrije Universiteit, Amsterdam, November 1997.

[4] Frank M. Brown, editor. *The frame problem in artificial intelligence: proceedings of the 1987 workshop, April 12-15, 1987, Lawrence, Kansas*. Kaufmann, 1987.

[5] E.M. Clarke, E.A. Emerson, and A.P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2), April 1986.

[6] R.B. Feenstra and R.J. Wieringa. LCM 3.0: a language for describing conceptual models. Technical Report IR-344, Faculty of Mathematics and Computer Science, Vrije Universiteit, Amsterdam, December 1993.

[7] Michael Gelfond and Vladimir Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, 17(2/3&4):301–321, 1993.

[8] Matthew L. Ginsberg and David E. Smith. Reasoning about action 2: the qualification problem. *Artificial Intelligence*, 35:311–342, 1988.

[9] E. Giunchiglia, G. N. Kartha, and V. Lifschitz. Actions with indirect effects (extended abstract). In *Proc. AAAI Spring Symposium'95 on Extending Theories of Actions*, 1995.

[10] F. Golshani, T.S.E. Maibaum, and M.R. Sadler. A modal system of algebras for database specification and query/update support. In *Proceedings of the Nineth International Conference on Very Large Databases*, pages 331–359, 1983.

[11] D. Harel. *First Order Dynamic Logic*. Springer, 1979. Lecture Notes in Computer Science 68.

[12] P. Jeremaes, S. Khosla, and T.S.E. Maibaum. A modal (action) logic for requirements specification. In D. Barnes and P. Brown, editors, *Software Engineering 86*, pages 278–294. Peter Peregrinus Ltd., 1986.

[13] S. Khosla and T.S.E. Maibaum. The prescription and description of state based systems. In B. Banieqbal, H. Barringer, and A. Pnueli, editors, *Temporal Logic in Specification*, pages 243–294. Springer, 1987. Lecture Notes in Computer Science 398.

[14] S. Khosla, T.S.E. Maibaum, and M. Sadler. Database specification. In T.B. Jr. Steel and R. Meersman, editors, *Database Semantics (DS-1)*, pages 141–158. North-Holland, 1986.

[15] V. Lifschitz. Two components of an action language. In *Annals of Mathematics and Artificial Intelligence*. 1997. to appear.

[16] F. Lin and R. Reiter. State Constraints Revisited. *Journal of Logic and Computation*, 4(5):655–678, 1994. Special Issue on Action and Processes.

[17] U. W. Lipeck and S. Brass. Object-oriented system specification using defaults. In H. Marburger K. von Luck, editor, *Management and Processing of Complex Data Structures - Third Workshop on Information Systems and Artificial Intelligence 1994*, Lecture Notes in Computer Science 777, pages 22–43. Springer-Verlag, 1994.

[18] K.L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, 1993.

[19] R. Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Vladimir Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: P apers in Honor of John McCarthy*. Academic Press, 1991.

[20] M. Ryan, J. Fiadeiro, and T. Maibaum. Sharing actions and attributes in modal action logic. In T. Ito and A.R. Meyer, editors, *Theoretical Aspects of Computer Software*, pages 569–593. Springer, 1991. Lecture Notes in Computer Science 526.

[21] R.J. Wieringa, J.-J. Ch. Meyer, and H. Weigand. Specifying dynamic and deontic integrity constraints. *Data and Knowledge Engineering*, 4:157–189, 1989.

[22] R.J. Wieringa and J.-J.Ch. Meyer. Actor-oriented specification of dynamic and deontic integrity constraints. In B. Talheim, J. Demetrovics, and H.-D. Gerhardt, editors, *3rd Symposium om Mathematical Fundamentals of Database and Knowledge Base Systems (MFDBS 91)*, pages 89–103. Springer, 1991. Lecture Notes in Computer Science 495. ftp://ftp.cs.vu.nl/pub/roelw/91-Actors.ps.Z. ftp://ftp.cs.vu.nl/pub/roelw/91-Actors.ps.Z.

[23] R.J. Wieringa, H. Weigand, J.-J. Ch. Meyer, and F. Dignum. The inheritance of dynamic and deontic integrity constraints. *Annals of Mathematics and Artificial Intelligence*, 3:393–428, 1991. ftp://ftp.cs.vu.nl/pub/roelw/91-Inheritance.ps.Z. ftp://ftp.cs.vu.nl/pub/roelw/91-Inheritance.ps.Z.

[24] M. Winslett. *Updating Logical Databases*. Cambridge University Press, 1990.