A modal logic for epistemic tests

Andreas Herzig and Jérôme Lang and Thomas Polacsek

Abstract. We study a modal logic of knowledge and action, focusing on test actions. Such knowledge-gathering actions increase the agents’ knowledge. We propose a semantics, and associate an axiomatics and a rewriting-based proof procedure.

1 Introduction

In order to be able to evolve in her/his environment, an intelligent agent must interact with the latter. For instance, imagine a robot that wants to open a door that might be locked. If the robot is shrewd enough, it starts by checking whether the door is effectively locked or not. This action enables it to know truth or falsehood of the formula “the door is locked”. We call test such an action.

For us, the concept of test is strongly attached to the concept of knowledge acquisition. There is a significant amount of related work about the interactions between action and knowledge. Combining knowledge and action in a logical framework comes back to the work of Moore [12] who provided a theory of action including knowledge-producing actions. Building on this theory, Scherl and Levesque [13] represent knowledge-producing actions in the situation calculus by means of an explicit accessibility relation between situations, treated as an ordinary fluent, that corresponds to our epistemic accessibility relation. Levesque [9] then uses this knowledge fluents to represent complex plans involving, like ours, observations and branching (and also loops, that we did not consider). He points out that the executability of a plan requires that the agent needs to know how to execute it, which implies that branching conditions must involve knowledge and not objective facts whose truth may not be accessible to the agent.

There are many domains where we learn the truth value of a given formula. Consider a logical circuit with six components $c_1, ..., c_6$. If the circuit is broken down, several configurations of faulty components are possible. Suppose that there are only three possible configurations: $\varphi_1$: components $c_1, c_2$ and $c_3$ are broken down (and only these ones); $\varphi_2$: components $c_1, c_2$ and $c_4$ are broken down (and only these ones); $\varphi_3$: components $c_1$ and $c_6$ are broken down (and only these ones).

Which test sequences enable us to know in which configuration we are? Testing components $c_2$ and $c_3$, for instance, enables us to know in which configuration we are; hence we should be able to prove that this sequence is sufficient.

A test is a particular interaction between the agent and the world. It is one-sided form of communication: the agent acquires knowledge about the environment, but does not change the environment. We make the following simplifying assumptions:

- the world does not evolve while the test is carried out;
- tests do not modify the environment;
- tests are reliable;
- every proposition can be tested successfully.

Database querying is a basic example of such tests.

What do we test? We can test objects of the world, for example a battery or a computer program. Here, we will be interested rather in testing facts: test if the battery is defective. A test plan is a (possibly conditional) sequence of tests. In our logic validating a test plan comes down to studying the validity of a logical formula.

Tests are actions. We shall integrate tests consequently into a logic of action. In order to be able to reason about the interactions between knowledge and actions, our logic comprises an epistemic component and a dynamic component.

In section 2 we give the necessary background on the logic EDL proposed in [6]. In Section 3 we extend EDL to EDL* so as to handle explicit tests of the form test whether a given formula is true or not; we give the axioms and the semantics of EDL*, and propose a rewriting procedure. In Section 4, we propose an alternative (and equivalent) definition of tests. Section 5 discusses related work.

2 The logic EDL

In [6] we presented a dynamic-epistemic logic called EDL for plan verification in the presence of partial observability. EDL allows for a large variety of actions (some of them change the world and some do not, some of them bring some new information and some do not, with all possible mixtures of these). EDL combines a fragment of propositional dynamic logic (PDL) and the epistemic logic $S5$. The language is built from a set of atomic formulas $FML_0$, a set of atomic actions $ACT_0$, the classical logic operators $\rightarrow, \land, \lor, \neg$, the action operators $\lambda$, if then else, “;”, the modal operator $[\cdot]$ and the epistemic operator $K$. We read the formula $\neg Kp \land \neg K\neg p$ “the agent neither knows $p$ nor $\neg p$”. As usual, $[s]A$ is an abbreviation of $\neg [s]\neg A$.

We say that a formula from $FML$ is objective if it contains no occurrence of $K$.

Our axiomatisation contains a modal logic for knowledge and a $K$ modal logic for actions. We take the logic of knowledge to be $S5$.

\begin{align*}
N(K) & \quad A \\
K(K) & \quad (KA \land K(A \rightarrow C)) \rightarrow KC \\
T(K) & \quad KA \rightarrow A \\
S(K) & \quad \neg KA \rightarrow K\neg KA \\
N(\alpha) & \quad [\alpha]A \equiv A \\
K(\alpha) & \quad ([\alpha]A \land [\alpha](A \rightarrow C)) \rightarrow [\alpha]C \\
Acq(\alpha) & \quad K[\alpha]C \rightarrow [\alpha]KC \\
Def(\lambda) & \quad [\lambda]A \equiv A \\
Def(\cdot) & \quad [s;\beta]A \equiv [s][\beta]A \\
Def(if) & \quad ([A \text{ then } \alpha \text{ else } \beta]A) \\
& \equiv ((A \rightarrow [\alpha]B) \land (\neg A \rightarrow [\beta]B))
\end{align*}
The semantics is in terms of possible states. We interpret the knowledge of the agent at a possible world $w$ by a set of worlds associated to $w$. Actions are interpreted as transition relations on worlds.

We define a model as a quadruple $M = \langle W, R_K, \{R_\alpha : \alpha \in ACT\}, V \rangle$ where $W$ is a set of possible worlds, $R_K \subseteq W \times W$ and every $R_\alpha \subseteq W \times W$ is an accessibility relation, and $V$ associates to each world an interpretation. We require

- $R_\alpha \{w\} = \{w\}$;
- $R_\alpha \beta = R_\alpha \circ R_\beta$;
- $\text{R if } A \text{ then } \alpha \text{ else } \beta = \{(w, w') \mid \text{if } \models_{M, w} A \text{ then } w R_\alpha w' \text{ else } w R_\beta w'\}$;
- $R_\alpha \circ R_K \subseteq R_K \circ R_\alpha$.

The truth conditions are defined as usual, in particular:

- $\models_{M, w} KA$ if $\models_{M, v} A$ for every state $v \in R_K(w)$;
- $\models_{M, w} [x]A$ if $\models_{M, w} A$ for every state $w' \in R_\alpha(w)$.

We call EDL our basic epistemic dynamic logic. EDL is sound and complete.

## 3 Adding explicit tests to EDL

### 3.1 EDL$^+$

Now we extend EDL to EDL$^+$ so as to include explicit tests. Such tests are actions of the form “test whether a given formula is true”, expressed by a new action operator: $?$. We read the formula $[A??]$ $\text{K}A$ “after testing if $A$, the agent knows $A$”. This language is very expressive since it allows for testing any formula, even formulas involving tests themselves, which leads to the possibility of expressing nested tests. However, practically, only elementary tests $A??$, where $A$ is modality-free, are relevant to AI applications such as diagnosis or decision making.

EDL$^+$ should be able to express for instance that after testing $c_3 \vee c_4$ and $c_3 \rightarrow c_4$, the truth value of $c_4$ is known. This means that the following formula $[c_3 \vee c_4??; c_3 \rightarrow c_4??](\text{K}c_4 \vee \text{K}\neg c_4)$ should be a theorem of EDL$^+$.

The language of EDL$^+$ is defined as follows. $FML$ and $ACT$ denote respectively the set of all well-formed formulas of EDL$^+$ and the set of all actions of EDL$^+$. These two sets being defined by mutual recursion:

- $FML$ is the smallest set containing
  - $p$ for all $p \in FML_0$;
  - $\neg A, A \lor B, A \land B, A \rightarrow B$ for all $A, B \in FML$;
  - $KA$ for all $A \in FML$;
  - $[\alpha]A$ for all $\alpha \in ACT$ and all $A \in FML$.

- $ACT$ is the smallest set containing
  - the empty action $\lambda$;
  - $\alpha$ for all $\alpha \in ACT_0$;
  - $\alpha;\beta$ for all $\alpha, \beta \in ACT$;
  - if $A$ then $\alpha$ else $\beta$ for all $A \in FML$ and all $\alpha, \beta \in ACT$;
  - $A??$ for all $A \in FML$.

In the following paragraph we give a list of axioms for tests. EDL$^+$ is obtained by gathering all these axioms and adding them to the basic EDL. However, this list should be understood as modular, namely, it is possible to build weaker logics by not including some of these axioms. Accordingly, we give each axiom together with its corresponding semantical condition.

### 3.2 Tests are ...

#### 3.2.1 ... purely informative

Purely informative actions never change the state of the world, they just change the knowledge. McIlraith [10] speaks about non-intrusive tests and van Linder et al. [11] about passive tests.

- $\text{Pl}(??) \quad C \rightarrow [A??]C$ if $C$ is an objective formula.

  Semantically, this corresponds to the condition
  - if $w_1, R_{A??} w_2$ then $V_{w_1} = V_{w_2}$

#### 3.2.2 ... deterministic and executable

Test are always executable and every execution in a given state always lead to the same result.

- $\text{Det}(??) \quad \langle A??\rangle C \leftrightarrow [A??]C$
- $\text{Exec}(??) \quad [A??]C \rightarrow \langle A??\rangle C$

  Both axioms can be gathered together into $\langle A??\rangle C \leftrightarrow [A??]C$.

  Semantically, $\text{Det}(??)$ and $\text{Exec}(??)$ correspond to the condition
  - $R_{A??}(w)$ is a singleton.

  The assumption that tests are always executable may look restrictive at first glance, because it may be the case that a given test has an applicability condition. We propose to express applicability conditions this way: if the test $A??$ has an applicability condition $\text{Cond}_A$ without which the execution of the test does not produce any result, we can replace $A??$ by “if $\text{Cond}_A$ then $A??$ else $\lambda$”.

#### 3.2.3 ... idempotent

Performing a test twice has no more effects than performing it once.

- $\text{A}(??) \quad [A??]C \rightarrow [A??][A??]C$

  Semantically, this corresponds to the transitivity of the accessibility relation $R_{A??}$:
  - if $w_1, R_{A??} w_2$ and $w_2, R_{A??} w_3$, then $w_1, R_{A??} w_3$.

**Proposition 1**

- The formula $[A??][A??]C \leftrightarrow [A??]C$ is provable from $\text{Det}(??), \text{Exec}(??)$ and $\text{Idem}(??)$.

#### 3.2.4 ... insensitive to negation

Testing whether a formula is true or not is equivalent to testing whether its negation is true or not:

- $\text{Neg}(??) \quad [A??]C \equiv [\neg A??]C$

  Semantically, this corresponds to the condition
  - $R_{A??} = R_{\neg A??}$.
3.2.5 ... reliable

We suppose that testing \( A \) when \( A \) is true always makes the agent knowing \( A \).

\[
\text{Reliab}(?) \rightarrow [A?] \top K A
\]

Semantically, this axiom corresponds to the condition

\[
\text{if } [\models_{M,w} A \text{ and } w(R_{\text{Alt}} \circ R_{\text{K}}) w' \text{ then } [\models_{M,w} A]
\]

**Proposition 2**

1. \([A?] (KA \vee K \neg A) \text{ is provable from Reliab}(?) \text{ and Neg}(?)\).
2. \([A?] (KA) \iff A \text{ is provable from Reliab}(?), \text{ Neg}(?) \text{ and Exec}(?)\).

(1) means that tests are always informative: after testing whether \( A \), the agent will always know the truth value of \( A \). Its proof goes as follows: using insensitivity to negation, we derive \( A \rightarrow [A?] (KA \vee \neg A) \), which together with \( A \rightarrow [A?] KA \) gives \([A?] (KA \vee KK A \neg A)\). (2) means that tests are correct, i.e., they never tell the agent a false statement. It is derived from \( A \rightarrow [A?] K A \) above and executability.

3.2.6 ... predictable

If \( A \) is true then testing if \( A \) eliminates exactly those accessible worlds where \( A \) is false. For Scherl and Levesque [13] this is the basic axiom for knowledge-gathering axioms.

\[
\text{Pred}(?, K) \rightarrow ([A?] KC \equiv K(A \rightarrow [A?] C)]
\]

Semantically, this corresponds to the condition

\[
\text{if } w R_{\text{Alt}} w' \text{ then } R_{\text{K}} (w') = \bigcup_{v \in R_{\text{K}} (w')} [\models_{M,w} A]
\]

**Proposition 3** \( L C \) be an objective formula. Then the proposition \( KC \rightarrow [A?]KC \) is provable in \( \text{EDL}^+ \) from \( P(I(??)) \) and \( Acq(\alpha_3), K \).

This proposition expresses that knowledge is reliable.

3.3 Example

Now we can handle our running example in our logic. A possible plan is \( A = c_3 > D; c_4 > D \) i.e. to first test component \( c_3 \) and then component \( c_4 \). We must prove that for every state of affairs, if the agent executes \( A \) then she knows the configuration of breakdown. In other words, we must prove that

\[
T \rightarrow [\pi](K \varphi_1 \vee K \varphi_2 \vee K \varphi_3)
\]

is a theorem of \( \text{EDL}^+ \), where

\[
T = \{ K((c_1 \vee C_2 \vee c_3 \vee c_4 \top c_5 \top c_6) \rightarrow \varphi_1), K((c_1 \vee C_2 \vee c_3 \vee c_4 \top c_5 \top c_6) \rightarrow \varphi_2), K((c_1 \vee c_2 \vee C_3 \vee c_4 \top c_5 \top c_6) \rightarrow \varphi_3), K(\varphi_1 \vee \varphi_2 \vee \varphi_3) \}
\]

\[
T \rightarrow [\pi](K \varphi_1 \vee K \varphi_2 \vee K \varphi_3) \text{ is provable in } \text{EDL}^+ \text{, which is intended.}
\]

3.4 Automated theorem proving

For the rest of the paper, we call models of \( \text{EDL}^+ \) the models of \( \text{EDL} \) satisfying all the semantical conditions of Section 3. The axiomatics of \( \text{EDL}^+ \) is that of \( \text{EDL} \) plus axioms of Section 3.

**Proposition 4** The following equivalences are theorems of \( \text{EDL}^+ \).

1. \([A]?((C_1 \land C_2) \equiv \top [A]?[C_1 \land [A]?[C_2])
2. \([A]?((C_1 \lor C_2) \equiv \top [A]?[C_1 \lor [A]?[C_2])
3. \([A]?[C] \equiv \neg[A]?[C]
4. \([A]?KC \equiv ((A \rightarrow K(A \rightarrow [A]?C)) \land (\neg A \rightarrow K(\neg A \rightarrow [A]?C))
5. \([A]?[C] \equiv C \text{ if } C \text{ is objective.}

**Proof:** 1 is provable with axiom \( K(\alpha_3) \), 2 with determinism \( Det(??) \), 3 with determinism and executability \( Exec(??) \). For 4 we need predictability \( Pred(??, K) \) and insensitivity to negation \( Neg(??) \). Finally for 5 we use \( Pl(??) \) and \( Det(??) \). \( \Box \)

Hence we can ‘push down’ the modal operator of test from the left to the right through all the other connectives \( K, \land, \lor, \neg \). When \([A]?[C] \) reaches an objective formula then we can apply \([A]?[C] \equiv C \) and thus eliminate one modal operator of test from the formula.

Iterating these rewrite steps we can obtain formulas without test operators. It is easy to prove that this rewriting system is confluent and terminating.

A consequence of Proposition 4 is that we can reduce the problem of proving theorems in our logic to that of proving theorems in \( S5 \). The reduction is done by rewrite rules.

**Proposition 5** \( \text{Suppose } ACT_0 = \emptyset. \text{ For every formula } A \text{ there exists a formula } A' \text{ without action operators such that } A \equiv A' \text{ is a theorem of } \text{EDL}^+ \).

3.5 Soundness and completeness

**Proposition 6** \( \text{EDL}^+ \) is sound.

**Proof:** Each of the axioms is valid, and the inference rules preserve validity. \( \Box \)

The above theorem gives us completeness for the fragment of \( \text{EDL}^+ \) where there are no other primitive actions than tests.

**Proposition 7** \( \text{Suppose } ACT_0 = \emptyset. \text{ Then } \text{EDL}^+ \) is complete.

**Proof:** Let \( A \) be consistent. According to theorem 5 of section 3.4 there exists a formula \( A' \) without action operators such that \( A \equiv A' \) is a theorem. Hence \( A' \) is consistent. Now \( A' \) is in the language of \( S5 \), and given that the axiomatics of our logic contains that of the epistemic logic \( S5 \), \( A' \) is as well consistent in \( S5 \). Via the complete-ness of \( S5 \) there must therefore exist a \( S5 \)-model containing a state \( w \) where \( A' \) is true. Then it is straightforward to extend that model to an \( \text{EDL}^+ \)-model where \( A' \) is true in \( w \). Finally, given that (due to soundness) the equivalences that we have used to rewrite formulas are valid, that model must also satisfy \( A \) in \( w \). \( \Box \)

4 An alternative test action

In this section we rewrite the test action as a nondeterministic composition of uninformative actions.
4.1 The logic $\text{EDL}_C^+$

We introduce into our langage a “check that” operator “??” mapping a formula to an action. We read $A$? as “establish $A$” or “check that $A$ is true”.

It is then possible to define the action $A$? of testing if $A$ as an abbreviation of the complex action of nondeterministic composition of “check that $A$” and “check that $\neg A$”. This corresponds to the axiom

$$\text{Def}(?) \quad [A?]C \equiv ([A]C \land [\neg A]C)$$

This formally expresses that testing whether $A$ is true or not amounts to nondeterministically choose between trying to establish that $A$ and trying to establish that $\neg A$.

The other way round we can choose ‘??’ to be primitive. In this case we can define ‘?’ by

$$\text{Def}(?) \quad [A?]C \equiv [A?][A \rightarrow C].$$

We give the following axiomatics for ‘?’.

$$\text{Det}(?) \quad [A?]C \rightarrow [A?]C$$

$$\text{PI}(?) \quad C \rightarrow [A?]C \text{ if } C \text{ is an objective formula}$$

$$\text{Exec}(?) \quad A \rightarrow [A?]T$$

$$\text{Relab}(?) \quad [A?]K/A$$

$$\text{Perm}(?) \quad \{A?\}KC \rightarrow K[A?]C$$

The logic $\text{EDL}_C^+$ is obtained by adding the five axioms above to the inference rules $N(A)$, $N(\alpha)$ and the axioms $K(K)$, $T(K)$, $K(\alpha)$, Def(\alpha), Def(\beta).

$$\text{Det}(?), \text{PI}(?), \text{Exec}(?), \text{Relab}(?), \text{Perm}(?) \text{ respectively stand for deterministic, purely informative, executability, reliability. The first three have}$$

- if $w(R_A \circ R_K)w'$ then $w' = (R_A \circ R_K)w$

The last axiom says that test ‘?’ are uninformative.

Proposition 8 $\text{EDL}_C^+$ plus Def(?) is equivalent to $\text{EDL}_C^+$ plus Def(??).

Proposition 9 $\text{EDL}_C^+$ is sound and complete.

4.2 Automated theorem proving and complexity

Proposition 10 The following equivalences are theorems of $\text{EDL}_C^+$.

1. $[A?]\neg C \equiv (A \rightarrow \neg [A?]C)$
2. $[A?]C \land C_2 \equiv ([A?]C_1 \land [A?]C_2)$
3. $[A?]C_1 \lor C_2 \equiv ([A?]C_1 \lor [A?]C_2)$
4. $[A?]KC \equiv ([A?]C \lor [K]A)C$
5. $[A?]C \equiv (A \rightarrow C)$ if $C$ is an objective formula

The proof of these equivalences is similar to that of proposition 4.

This result implies that the fragment of $\text{EDL}_C^+$ where tests are non-nested has a complexity not higher than the complexity of epistemic logic. Indeed, in case we rewrite our procedure is a polynomial transformation into the epistemic logic. Hence the problem of deciding theoremhood in that fragment has the same complexity as in epistemic logic.

The problem of deciding whether a given S5-formula is a theorem is coNP-complete, it follows that

$\text{Proposition 11} \quad \text{The validity problem in the fragment of } \text{EDL}_C^+ \text{ where (i) } \text{ACT}_0 = \emptyset \text{ and (ii) only non-nested tests are considered, is coNP-complete.}$

5 Related work

Several logics of knowledge and action exist in the literature. Closest to ours is the work of Gerbrandy and Groeneveld [2, 4, 5]. Their Dynamic Epistemic Logic has two sorts of tests, the first of which is denoted by $\text{?A}$ and is the standard dynamic logic test: “it succeeds […] when $A$ is true, and fails otherwise”. Consequently $[\text{?A}]C$ is an abbreviation of $A \rightarrow C$. The second one is noted $\text{U}\alpha$ and “corresponds to [the] agent […] learning that program $\alpha$ has been executed”. (We have slightly adapted notations.) This means that agents act a priori unconsciously and must explicitly learn about the executions of their actions.$^4$

$\text{U}\alpha A$ is similar to our $A$. More precisely, our logic can be mapped into Gerbrandy’s logic [2]: our action $A$ can be translated into their $A; U\alpha A$.

In [2] an axiomatics is given, which is similar to ours. Nevertheless there are subtle differences. We already mentioned the first one: there is a non-epistemic test ‘?A supplementing the epistemic test $U\alpha A$.

The second main difference is that there, the logic of knowledge is $K$, while ours is at least $S3$. Hence there is no axiom $T(K)$. It seems to be problematic to add these axioms to the logic. This will be detailed after our next point.

The third main difference is that there, instead of axiom $\text{Exec}(?) A \rightarrow [A?]\neg A$ there is an equivalence $A \equiv [\neg A]A$ (axiom 5 in [2]). This means that an agent can always successfully learn about the execution of test. This leads to difficulties at least if we suppose that the epistemic notion under concern satisfies a consistency requirement - as expressed by the modal axiom $D(K) K\alpha \rightarrow [\neg K\neg A]$ (that is a consequence of axiom $T(K)$). Indeed, suppose $p$ is an atom. Then $[U\alpha p]Kp$ is derivable in their logic, as well as $K \rightarrow [U\alpha]K\rightarrow p$. But from these two we can derive $K \rightarrow [U\alpha p] (K(p \land \neg p)$. While in our logic this means that the test action fails, in theirs the test $U\alpha p$ always succeeds. Therefore axiom $D(K)$ cannot be added to their logic as it stands.

Finally a more technical difference are the respective completeness proofs. While ours basically uses a reduction to a modal logic without tests, theirs is a (much longer) Henkin type proof. Nevertheless, our technique also applies to their logic, and permits thus to obtain a much simpler proof. To witness, the $K$-axiom for $[U\alpha]$ together with the above equivalence $A \equiv [U\alpha]A$ pass through the modal operator $[U\alpha]$ through conjunction, disjunction, and negation, and their axiom 7 $[U\alpha]K\alpha \equiv [K\alpha][U\alpha]A$ allows to pass through the epistemic operator $K$. Finally, their axiom 6 permits to eliminate the $[U\alpha]$ operator from formulas. Thus one can follow the same line of reasoning as in our completeness proof. This has been done in [3].

In a series of articles Segerberg has developed a logic of belief and action called Doxastic Dynamic Logic (DDL) [15, 14]. There are three types of modalities $+A$, $-A$, and $\alpha$ the first of which corresponds to our $A$. He discusses axioms for $+A$ that are similar to ours, but nevertheless closer to Gerbrandy and Groeneveld’s work.

$^4$ The authors consider several agents and groups of agents. We abstract from that here.

$^5$ This leaves more flexibility than our language, for instance they can express that agent $i$ learns that agent $j$ learned that $A$ has been tested (by $U_i U_j A$).

$^6$ This makes it also possible to write the preservation axiom PI(?) as an equivalence.
To witness, he also considers that tests are always executable and deterministic, i.e. he has the axiom \([+A] \neg C \equiv \neg [+A] C\) (his axiom 13), as well as a preservation axiom in terms of equivalence (his axiom 10). Therefore our above remarks also apply to this approach.

Van Linder et al. [11] propose two formalizations of the concept of test, respectively in S5 and KT. They show that the tests can be seen on the one hand as operators of epistemic update, and on the other hand as operators of knowledge expansion. In both cases, the knowledge of the agent after the execution of a test is obtained by minimizing the changes between the ol and the new state of knowledge.

Another line of research has been developed in the AI field of reasoning about actions around the concept of knowledge gathering actions [13, 9, 7, 10] (see [6]).

The logic AOL of Lakemeyer and Levesque has similarities with EDL. The main difference is that our logic does not contain the concept of only knowing. To witness we consider an example given in their paper. “Suppose we have a robot that knows nothing about the initial state of the environment, but that there is a sensing action, reading a sonar, which tells the robot when it is getting close to a wall.” Let us read the atomic formulas \(c\) and \(s\) respectively as ‘the robot is close to the wall’ and ‘the sonar works’, and let us interpret \(mc\) and \(ma\) respectively as the atomic actions of moving closer and moving away from the wall. In our language, what they then want to prove is

\[
\begin{align*}
1. \quad & [c?][Kc \lor K\neg c] \\
2. \quad & Ku \rightarrow [c?][Kc \lor K\neg c] \\
3. \quad & K\neg c \rightarrow [mc](\neg Kc \land \neg K\neg c) \\
4. \quad & K\neg c \rightarrow [mc](\neg Kc \land \neg K\neg c)
\end{align*}
\]

Only the last formula requires the non-monotonic only knowing notion.

Lastly, let us mention two approaches that investigate the properties of tests in pure propositional logic. McIlraith [10] gives an abductive characterization of tests enabling to classify according to their discriminating power. Lang and Marquis [8] investigate the complexity of test planning and give an algorithm which generates all minimal sets of tests enabling discriminating among a set of competing hypotheses.

6 Conclusion

We have proposed a logic of knowledge and action, EDL \(^+\), which focuses on the notion of test. Two equivalent formalizations are given. The first one, which is more intuitive, uses tests of the type ‘??’ (“test if”). The second one uses tests of the type ‘?’ (“check that”). Expressing “check that” using “test if” is easy. The transformation is linear in time and space. The other way round is more complex: expressing “test if” using “check that” tests as primitive actions using the rewriting rule \(A?? \rightarrow A? \lor (\neg A)?\) may be spatially exponential.

In [6], we have shown that the full logic EDL (of which EDL \(^+\) is a fragment) enables us expressing plan verification as a validity problem in EDL. This also applies here for the fragment consisting of tests only, which means that EDL \(^+\) can be successively used for verifying that a given test plan (linear or conditional) enables reaching an epistemic goal (such as discriminating among a set of competing diagnoses).

We plan to continue that work in two directions. First, while our logic allows to reason about the evolution of knowledge by tests and allows for plan verification it cannot find test plans. This might be achieved in a way similar to the approach in [1]. Second, it would be interesting to bring closer our work to logics of dynamic logics of probability so as to handle stochastic effects of actions. This would be relevant for modelling unreliable tests whose probability of reliability is known.

REFERENCES


