Making Things Happen

Jan van Eijck (jve@cwi.nl)
CWI, Amsterdam, ILLC, Amsterdam, Uil-OTS, Utrecht

Abstract. We explore some logics of change, focusing on commands to change the world in such a way that certain elementary propositions become true or false. This investigation starts out from the following two simplifying assumptions: (1) the world is a collection of facts (Wittgenstein), and (2), the world can be changed by changing elementary facts (Marx). These assumptions allow us to study the logic of imperatives in the simplest possible setting.

Keywords: semantics of natural language, modal logic, dynamic logic, knowledge representation languages.

1991 CR Subject Classification: F.3.1, F.3.2, I.2.4, I.2.7.

1. The Logic of Action Without Repercussion

In natural language, the distinction between imperative mode and declarative mode is made by assuming that declarative sentences describe a state of the world, while imperative sentences convey an intention of the speaker that the addressee takes responsibility for changing the world in some particular way. We will study some simple logical languages where commands to change the world are interpreted literally as transitions that make things happen by effecting the desired change.

If one assumes that the world is just a collection of unconnected facts, and that elementary changes to the world can be made independently of each other then a logic of change can look very simple. Changing a world w by making p true (false) results in a world v which is just like w except for the fact that p is true (false) in it. Notation for this: $w = v(p|1)$ or $w = v(p|0)$.

Looking at the commands to make p true or false as modalities, the following is an appropriate language for the logic of independent change:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid [\boldsymbol{\pi}]\varphi.$$

$$\pi ::= p := 1 \mid p := 0 \mid \pi;\pi \mid \pi \cup \pi \mid \varphi?.$$

We employ the usual abbreviations for $\top$, $\bot$, $\varphi \vee \psi$, $\varphi \rightarrow \psi$, $\varphi \leftrightarrow \psi$ and $\langle\boldsymbol{\pi}\rangle\varphi$.

If one assumes that every valuation is reachable, then there is only one model M, namely $M = \{0,1\}^P$ (the set of all valuations for P), and we can suppress the parameter M from the semantic clauses in the definition of $w \models \varphi$:

1. Clauses for atoms and Booleans: as usual.
2. $w \models [\boldsymbol{\pi}]\varphi$ iff for all v with $w, v \models \pi$ it is the case that $v \models \varphi$.
3. $w, v \models p := 1$ iff $v = w(p|1)$.
4. $w, v \models p := 0$ iff $v = w(p|0)$.
5. $w, v \models \pi_1; \pi_2$ iff there is an x with $w, x \models \pi_1$ and $x, v \models \pi_2$.
6. $w, v \models \pi_1 \cup \pi_2$ iff either $w, v \models \pi_1$ or $w, v \models \pi_2$.
7. $w, v \models \varphi?$ iff $v = w$ and $w \models \varphi$.

The notion $\models \varphi$ is defined as: for every valuation v, $v \models \varphi$. $\Gamma \models \varphi$ means: for every valuation v with $v \models \gamma$ for all $\gamma \in \Gamma$, it holds that $v \models \varphi$.

We show that the notion $\models \varphi$ for this language is very simple indeed. Note first that the requirement of bisimulation equivalence for the logic of action without repercussion boils down to 'having the same valuation', and the zig-zag requirement trivializes. It follows from this that the global notion of validity for the logic of propositional action without repercussion is not very exciting. Formulas of this logic have precisely the same descriptive power as formulas of propositional logic, so the valid formulas of the logic of propositional action without repercussion are precisely the formulas equivalent to propositional tautologies.

Still, it is illuminating to take a further look at this logic. In the following, we use B as a meta-variable for $1, 0$, so we can say that both $p := 1$ and $p := 0$ are of the form $p := B$. Also, we need the following notion of substitution:

DEFINITION 1. (Substitution). *$[\psi/p]\varphi$ is given by:*

$$
\begin{array}{lcl}
[\psi/p]q & := & \begin{cases} \psi & \textit{if } p = q. \\ q & \textit{otherwise} \end{cases} \\
[\psi/p](\neg\varphi) & := & \neg[\psi/p]\varphi \\
[\psi/p](\varphi_1 \wedge \varphi_2) & := & [\psi/p]\varphi_1 \wedge [\psi/p]\varphi_2 \\
[\psi/p]([\boldsymbol{q := B}]\varphi) & := & \begin{cases} [\boldsymbol{q := B}][\psi/p]\varphi & \textit{if } p \neq q. \\ [\boldsymbol{q := B}]\varphi & \textit{otherwise} \end{cases} \\
[\psi/p]([\boldsymbol{\pi_1; \pi_2}]\varphi) & := & [\psi/p]([\boldsymbol{\pi_1}][\boldsymbol{\pi_2}])\varphi \\
[\psi/p]([\boldsymbol{\pi_1 \cup \pi_2}]\varphi) & := & [\psi/p]([\boldsymbol{\pi_1}]\varphi) \vee [\psi/p]([\boldsymbol{\pi_2}]\varphi) \\
[\psi/p]([\boldsymbol{\varphi_1?}]\varphi_2) & := & [\boldsymbol{(}[\boldsymbol{\psi/p}]\boldsymbol{\varphi_1)?}][\psi/p]\varphi_2.
\end{array}
$$

A translation procedure for this language into propositional logic can now be given, as follows:

$$
\begin{array}{rcl}
p^{\odot} & := & p \\
(\neg\varphi)^{\odot} & := & \neg\varphi^{\odot} \\
(\varphi_1 \wedge \varphi_2)^{\odot} & := & \varphi_1^{\odot} \wedge \varphi_2^{\odot} \\
([\boldsymbol{p := 1}]\varphi)^{\odot} & := & [\top/p]\varphi^{\odot} \\
([\boldsymbol{p := 0}]\varphi)^{\odot} & := & [\bot/p]\varphi^{\odot} \\
([\boldsymbol{\varphi?}]\psi)^{\odot} & := & \varphi^{\odot} \rightarrow \psi^{\odot} \\
([\boldsymbol{\pi_1; \pi_2}]\varphi)^{\odot} & := & ([\boldsymbol{\pi_1}][\boldsymbol{\pi_2}]\varphi)^{\odot} \\
([\boldsymbol{\pi_1 \cup \pi_2}]\varphi)^{\odot} & := & [\boldsymbol{\pi_1}]\varphi^{\odot} \wedge [\boldsymbol{\pi_2}]\varphi^{\odot}
\end{array}
$$

It is easy to check that this translation is correct, in the sense that $v \models \varphi$ iff $v \models \varphi^{\odot}$.

2. Application: Database Updating

Spruit, Wieringa and Meyer (1995) investigate the logic of passive and active insertions into a propositional database. Their database actions turn out to be expressible in our action logic.

- passive insertion of p: corresponds to $p := 1$,
- passive deletion of p: corresponds to $p := 0$,
- active insertion of p with re-computation φ: corresponds to $p := 1; (p \wedge \varphi)^{\bullet}$,
- active deletion of p with re-computation φ: corresponds to $p := 0; (\neg p \wedge \varphi)^{\bullet}$.

Notation for passive insertion of p: I_p, for passive deletion of p: D_p. Notation for active insertion of p with re-computation φ: I_p^{φ}, for active deletion of p with re-computation φ: D_p^{φ}. Intuitively, $(p \wedge \varphi)^{\bullet}$ denotes the command that makes $(p \wedge \varphi)$ true 'in a minimal way', i.e., by making the smallest possible number of atomic changes. The translation instruction $^{\bullet}$ for transforming re-computation formulas into appropriate commands will be given below.

In (Spruit, Wieringa and Meyer, 1995) the re-computations φ are propositional formulas in Horn clause form. In fact, there is no harm in allowing re-computations to be an arbitrary propositional formulas, subject to the following conditions:

- a re-computation φ is not a contradiction,
- if φ is a re-computation after insertion of p, then the clausal form of φ,
$$\bigwedge(r_1 \wedge \cdots \wedge r_n \rightarrow s_1 \vee \cdots \vee s_m)$$
has to satisfy: if $p = r_i$ then $m > 0$,
- if φ is a re-computation after deletion of p, then the clausal form of φ,
$$\bigwedge(r_1 \wedge \cdots \wedge r_n \rightarrow s_1 \vee \cdots \vee s_m)$$
has to satisfy: if $p = s_j$ then $n > 0$.

The intuitive idea is to use φ to do a minimal re-computation, i.e., to make the database comply with φ by changing a minimal number of facts.

Examples: $I_p^{p\rightarrow q}$ corresponds to $p := 1; q := 1$. $D_p^{q\rightarrow p}$ corresponds to $p := 0; q := 0$.

Here is how to get these correspondences in a systematic way. Suppose we start with I_p^{φ}. Because of the conditions on φ in the I_p^{φ} case, $\varphi \wedge p$ is never a contradiction. Transform $\varphi \wedge p$ into a command $(\varphi \wedge p)^{\circ}$ as follows. First put $\varphi \wedge p$ in dual clause form. Replace disjunction signs between dual clauses by occurrences of $\cup$, and replace each dual clause $r_1 \wedge \cdots \wedge r_n \wedge \neg s_1 \wedge \cdots \neg s_m$ by $\perp$? if $r_i = s_j$ for some i, j, and by $r_1 := 1; \cdots ; r_n := 1; s_1 := 0; \cdots ; s_m := 0$ otherwise.

It is clear that the procedure $(\varphi \wedge p)^{\circ}$ makes $\varphi \wedge p$ true. Next, from $(\varphi \wedge p)^{\circ}$ we have to get at a procedure $(\varphi \wedge p)^{\bullet}$ that makes $\varphi \wedge p$ true in a minimal way. More generally, suppose we add a command operator μ to the language, with semantics given by $v, w \models \mu\pi$ iff $v, w \models \pi$ and there is no u with $v, u \models \pi$ and u differs from v in less atomic facts than w does.

Note that μ minimization is different from minimization in preferential reasoning, where μ is a program in its own right that picks out the set of all most preferred elements from an index set of possible worlds. This notion is then further analyzed in terms of a modality $[<]$, with $[<]\varphi$ interpreted as 'φ is true in all more $<$ preferred worlds. In the current setting, such an approach does not work, for the most $<$ preferred world without further ado is always the current world itself.

As an example, here is a definition of $\mu\pi$ for the case where π equals $p := 1 \cup q := 1$. Abbreviate $\varphi?; \pi_1 \cup (\neg\varphi)?; \pi_2$ as
$$\text{IF } \varphi \text{ THEN } \pi_1 \text{ ELSE } \pi_2.$$
Then we have:
$$\mu(p := 1 \cup q := 1) = \text{ IF } \neg p \wedge \neg q \text{ THEN } p := 1 \cup q := 1 \text{ ELSE } \top?$$

If either p or q is already the case, nothing happens, so it is impossible to end up in a situation where both p and q are the case unless $p \wedge q$ already holds in the initial state.

The example illustrates that in case of a choice involving a (finite) set of proposition letters S we can always enumerate the possible values of S and determine what should be done in each individual case to minimize the number of changes. More specifically, note that each command can be written in the form

$$\bigcup t_1; \cdots; t_n; a_1; \cdots a_m,$$

where

- all the t_i are tests on different literals,
- all the a_j are atomic actions on different proposition letters,
- if q occurs in both $t_1; \cdots; t_n$ and $a_1; \cdots; a_m$ then either in the form $\cdots q? \cdots q := 0 \cdots$ or in the form $\cdots \neg q? \cdots q := 1 \cdots$.

Write a command π in expanded form as $\bigcup_i T_1; A_i$. List all descriptions for the possible valuations to the proposition letters in π as $\varphi_1, \ldots, \varphi_k$. Now $\mu(\pi)$ has the form:

$$\begin{array}{l}
\text{IF } \varphi_1 \text{ THEN } B_1 \\
\text{ELSE IF } \cdots \\
\vdots \\
\text{ELSE IF } \varphi_k \text{ THEN } B_k \\
\text{ELSE } \top?.
\end{array}$$

Here each B_j is the union of all shortest lists A_i such that $T_i; A_i$ is in the expansion of π, and $\varphi_j \models \langle \boldsymbol{T_i} \rangle \top$ (test T_i succeeds on the valuation given by φ_j). This shows that μ is a definable operator.

Here is an example application of this procedure, for the case of $\mu((p := 1; q := 1) \cup (r := 1; s := 1))$.

$$\begin{array}{l}
\mu((p := 1; q := 1) \cup (r := 1; s := 1)) = \\
\quad \text{IF } \neg p \wedge \neg q \wedge \neg r \wedge \neg s \text{ THEN } (p := 1; q := 1) \cup (r := 1; s := 1)) \\
\quad \text{ELSE IF } p \wedge \neg q \wedge r \wedge \neg s \text{ THEN } q := 1 \cup s := 1 \\
\quad \text{ELSE IF } p \wedge \neg q \wedge \neg r \wedge s \text{ THEN } q := 1 \cup r := 1 \\
\quad \text{ELSE IF } \neg p \wedge q \wedge r \wedge \neg s \text{ THEN } p := 1 \cup s := 1 \\
\quad \text{ELSE IF } p \wedge \neg q \wedge \neg r \wedge s \text{ THEN } p := 1 \cup r := 1 \\
\quad \text{ELSE IF } p \wedge \neg q \wedge \neg r \wedge \neg s \text{ THEN } q := 1 \\
\quad \text{ELSE IF } \neg p \wedge q \wedge \neg r \wedge \neg s \text{ THEN } p := 1 \\
\quad \text{ELSE IF } \neg p \wedge \neg q \wedge r \wedge \neg s \text{ THEN } s := 1 \\
\quad \text{ELSE IF } \neg p \wedge \neg q \wedge \neg r \wedge s \text{ THEN } r := 1 \\
\quad \text{ELSE } \top?
\end{array}$$

Returning to the database operations that motivated the analysis, we define I_p^φ as $p := 1; \mu(\varphi \wedge p)^\circ$, and and D_p^φ as $p := 0; \mu(\varphi \wedge \neg p)^\circ$.

3. Imperatives on Their Own

So far, we have talked about the command language in close connection with the associated propositional assertion language. But it is also possible to study the commands on their own. The following language gives the set of commands that we have discussed.

$$\pi ::= p? \mid \neg p? \mid p := 1 \mid p := 0 \mid \pi;\pi \mid \pi \cup \pi.$$

The intended semantics is implicit in the semantic clauses given above. We write $[\![\pi]\!]$ for $\{(x,y) \mid x,y \models \pi\}$. A reasonable consequence relation for this language is $\pi_1 \models \pi_2$ iff rng $([\![\pi_1]\!]) \subseteq \text{dom}([\![\pi_2]\!])$. A command π_1 has another command π_2 as a consequence if performing π_1 sets up the conditions under which π_2 can be performed. Special case: $\models \pi$ iff $\text{dom}([\![\pi]\!]) = \{0,1\}^P$. A command π is valid if it can always be performed.

Note that our dynamic command consequence notion behaves quite differently from static consequence notions. Command repetition is not harmless: $p?;p := 0 \not\models p?;p := 0$, and monotonicity does not hold without further ado, for we have $p? \models p?$ but $p?;p := 0 \not\models p?$

To investigate this language, define functions $\varphi[+p]$ and $\varphi[-p]$ for positive and negative substitutions of proposition letters in propositional formulas, as follows.

$$\begin{array}{lcllcl}
p[+p] & := & p & p[-p] & := & \neg p \\
q[+p] & := & q & q[-p] & := & q \\
(\neg\varphi)[+p] & := & \neg(\varphi[-p]) & (\neg\varphi)[-p] & := & \neg(\varphi[+p]) \\
(\varphi \wedge \psi)[+p] & := & (\varphi[+p] \wedge \psi[+p]) & (\varphi \wedge \psi)[-p] & := & (\varphi[-p] \wedge \psi[-p])
\end{array}$$

Next, define a 'next state' function $NS(\varphi, \pi)$ which computes the next state condition of every command π, given a formula of propositional logic φ specifying the initial state (the definition is in the spirit of Van Benthem (1993)):

$$\begin{array}{lcl}
NS(\varphi, p?) & := & \varphi \wedge p \\
NS(\varphi, \neg p?) & := & \varphi \wedge \neg p \\
NS(\varphi, p := 1) & := & \varphi[+p] \\
NS(\varphi, p := 0) & := & \varphi[-p] \\
NS(\varphi, \pi_1;\pi_2) & := & NS(NS(\varphi, \pi_1), \pi_2) \\
NS(\varphi, \pi_1 \cup \pi_2) & := & NS(\varphi, \pi_1) \vee NS(\varphi, \pi_2).
\end{array}$$

It is easy to see that $w \models \varphi$ and $w, v \models \pi$ iff $v \models NS(\varphi, \pi)$. It follows that $\pi_1 \models \pi_2$ iff $\text{rng}\,([\![\pi_1]\!]) \subseteq \text{dom}([\![\pi_2]\!])$ iff $NS(NS(\top, \pi_1), \pi_2)$ is satisfiable, and $\models \pi$ iff $NS(\top, \pi)$ is satisfiable. Thus a translation argument via NS demonstrates that, once again, we are just doing propositional logic.

Still, it is illuminating to give a direct axiomatization, without resorting to an assertion language. We employ sequent format, with sequents of the form $X \Longrightarrow \pi$, where X is a finite list of procedures from the language.

Structural rules are left monotonicity and compositional cut (well known from the structural analysis of dynamic inference modes in Van Benthem (1996)), plus restricted forms of contraction, weakening and permutation. Since ; is clearly associative, we do not bother to write brackets in $\pi_1; \pi_2; \pi_3$, and we dispense with a rule that spells out the equivalence between $(\pi_1; \pi_2); \pi_3$ and $\pi_1; (\pi_2; \pi_3)$. We use $A?$ to refer to a test of the form $p?$ or $\neg p?$.

Axioms:

$$p := 1 \Longrightarrow p? \qquad\qquad p := 0 \Longrightarrow \neg p? \qquad\qquad A? \Longrightarrow A?$$

Left monotonicity:

$$\frac{X \Longrightarrow \pi}{YX \Longrightarrow \pi}$$

Cut with Composition-introduction:

$$\frac{X \Longrightarrow \pi_1 \quad \pi_1 \Longrightarrow \pi_2}{X \Longrightarrow \pi_1; \pi_2}$$

Test introduction:

$$\frac{X \Longrightarrow A? \quad X \Longrightarrow \pi}{X \Longrightarrow A?; \pi}$$

$\cup$ introduction:

$$\frac{X \Longrightarrow \pi_i}{X \Longrightarrow \pi_1 \cup \pi_2}\ i = 1, 2$$

$\cup$ elimination:

$$\frac{X, \pi_1 \Longrightarrow \pi \qquad\qquad X, \pi_2 \Longrightarrow \pi}{X, \pi_1 \cup \pi_2 \Longrightarrow \pi}$$

Test and assignment contraction rules:

$$\frac{Xp := 1Y \Longrightarrow \pi \quad Xp := 0Y \Longrightarrow \pi}{XY \Longrightarrow \pi}$$

$$\frac{XA?A?Y \Longrightarrow \pi}{XA?Y \Longrightarrow \pi} \qquad \frac{Xp := B_1; p := B_2 Y \Longrightarrow \pi}{Xp := B_2 Y \Longrightarrow \pi}$$

$$\frac{Xp?, p := 1Y \Longrightarrow \pi}{Xp?Y \Longrightarrow \pi} \qquad \frac{X\neg p?, p := 0Y \Longrightarrow \pi}{X\neg p?Y \Longrightarrow \pi}$$

$$\frac{Xp := 1, p?Y \Longrightarrow \pi}{Xp := 1Y \Longrightarrow \pi} \qquad \frac{Xp := 0, \neg p?Y \Longrightarrow \pi}{Xp := 0Y \Longrightarrow \pi}$$

Swap rules:

$$\frac{Xp := B_1, q := B_2 Y \Longrightarrow \pi}{Xq := B_2, p := B_1 Y \Longrightarrow \pi}\ p \neq q \qquad \frac{XA?, B?Y \Longrightarrow \pi}{XB?, A?Y \Longrightarrow \pi}$$

$$\frac{Xp := B, \pm q?Y \Longrightarrow \pi}{X \pm q?, p := BY \Longrightarrow \pi}\ p \neq q$$

$$\frac{X \pm p?, q := BY \Longrightarrow \pi}{Xq := B, \pm p?Y \Longrightarrow \pi}\ p \neq q$$

Here are some example derivations:

$$\frac{\Longrightarrow p := 1 \qquad\qquad p := 1 \Longrightarrow p?}{\Longrightarrow p := 1; p?}$$

$$\frac{\dfrac{p := 1 \Longrightarrow p?}{p := 1 \Longrightarrow p? \cup \neg p?} \quad \dfrac{p := 0 \Longrightarrow \neg p?}{p := 0 \Longrightarrow p? \cup \neg p?}}{\Longrightarrow p? \cup \neg p?}$$

$$\frac{\dfrac{p? \Longrightarrow p?}{q?, p? \Longrightarrow p?} \quad \dfrac{\dfrac{q? \Longrightarrow q?}{q?, p? \Longrightarrow q?}}{p?, q? \Longrightarrow q?}}{q?, p? \Longrightarrow p?; q?}$$

It is easy to check that the axioms and rules are sound. To see that the axiomatization is also complete, note that the axioms and rules encode the principles underlying the analysis of the previous section.

The notion $\pi_1 \models \pi_2$ that we have analyzed is called 'update-to-domain' consequence in (Van Benthem, 1996). One might ask whether there are other reasonable notions of 'command consequence'? Here are some candidates:

– $\pi_1 \models_2 \pi_2$ iff $[\![\pi_1; \pi_2]\!] \neq \emptyset$ (command π_1 logically implies command π_2 if at least one outcome of π_1 sets up the conditions for performing π_2).

– $\pi_1 \models_3 \pi_2$ iff $[\![\pi_1]\!] = [\![\pi_1; \pi_2]\!]$. (command π_1 logically implies command π_2 if performing π_1 makes execution of π_2 superfluous).

These possibilities are additions to the list of dynamic consequence notions that is discussed in (Van Benthem, 1996).

4. Action Without Repercussion over Subset Models

A subset model M is a non-empty subset of $\{0,1\}^P$. Interpret the update commands as follows in M:

$$\begin{aligned} [\![p := 1]\!]^M &:= \{\langle v, w\rangle \mid v, w \in M, w = v(p|1)\} \\ [\![p := 0]\!]^M &:= \{\langle v, w\rangle \mid v, w \in M, w = v(p|0)\} \end{aligned}$$

We now have a genuine modal logic. Its axiomatization (in standard Hilbert format, this time) involves the following notion.

DEFINITION 2. (p occurs freely in φ).
p occurs freely in p,
p occurs freely in $\neg\varphi$ iff p occurs freely in φ,
p occurs freely in $\varphi_1 \wedge \varphi_2$ iff p occurs freely in φ_1 or p occurs freely in φ_2,
p occurs freely in $[\boldsymbol{Q := B}]\varphi$ iff $p \neq q$ and p occurs freely in φ,
p occurs freely in $[\boldsymbol{\pi_1;\pi_2}]\varphi$ iff p occurs freely in $[\boldsymbol{\pi_1}][\boldsymbol{\pi_2}]\varphi$,
p occurs freely in $[\boldsymbol{\pi_1 \cup \pi_2}]\varphi$ iff p occurs freely in $[\boldsymbol{\pi_1}]\varphi$ or p occurs freely in $[\boldsymbol{\pi_2}]\varphi$,
p occurs freely in $[\boldsymbol{\varphi_1?}]\varphi_2$ iff p occurs freely in φ_1 or p occurs freely in φ_2.

The axiom schemas that are needed are:

- all substitution instances of propositional tautologies,
- the K schema for the atomic assignment modalities:

$$[p := B](\varphi \to \psi) \to ([p := B]\varphi \to [p := B]\psi).$$

- axioms that express the effect of $p := B$ on the value of p:

$$[\boldsymbol{p := 1}]p \qquad\qquad [\boldsymbol{p := 0}]\neg p.$$

- the schema of independent change for atomic assignment modalities:

$$\varphi \to [\boldsymbol{p := B}]\varphi$$

  provided p does not occur freely in φ.

– substitution schemas for atomic assignment modalities:

$$[\boldsymbol{p := 1}]\varphi \to [\boldsymbol{p := 1}]\bot \vee [\top/p]\varphi$$

$$[\boldsymbol{p := 0}]\varphi \to [\boldsymbol{p := 0}]\bot \vee [\bot/p]\varphi$$

– determinism axiom:

$$\langle \boldsymbol{p := B} \rangle\varphi \to [\boldsymbol{p := B}]\varphi.$$

– axioms for command decomposition:

$$[\boldsymbol{\varphi ?}]\psi \leftrightarrow (\varphi \to \psi).$$

$$[\boldsymbol{\pi_1 ; \pi_2}]\varphi \leftrightarrow [\boldsymbol{\pi_1}][\boldsymbol{\pi_2}]\varphi.$$

$$[\boldsymbol{\pi_1 \cup \pi_2}]\varphi \leftrightarrow [\boldsymbol{\pi_1}]\varphi \wedge [\boldsymbol{\pi_2}]\varphi.$$

– Modus Ponens:

$$\frac{\varphi \quad \varphi \to \psi}{\psi}$$

– Necessitation for atomic assignment commands:

$$\frac{\varphi}{[\boldsymbol{p := B}]\varphi}$$

Theorems are all formulas that can be derived from the axioms by a finite number of applications of the rules.

Note that $\bot \to [\boldsymbol{p := B}]\bot$ is a substitution instance of the schema of independent change. Therefore we have that $\langle \boldsymbol{p := B} \rangle\top$ is a theorem of the calculus. This allows us to strengthen the determinism schema as follows:

$$\langle \boldsymbol{p := B} \rangle\varphi \leftrightarrow [\boldsymbol{p := B}]\varphi.$$

Note also that if p does not occur freely in φ, then $[\top/p]\varphi = [\bot/p]\varphi = \varphi$, so we can strengthen the schema of independent change to an equivalence, as follows:

$$\varphi \leftrightarrow [\boldsymbol{p := B}]\varphi$$

provided p does not occur freely in φ.

Similarly, p does not occur freely in $[\top/p]\varphi$ or $[\bot/p]\varphi$, so we get from the schema of independent change that $[\top/p]\varphi \to [\boldsymbol{p := 1}]\varphi$, and $[\bot/p]\varphi \to [\boldsymbol{p := 0}]\varphi$. Also, since $\bot \to \varphi$ is a propositional tautology, we get by necessitation $[\boldsymbol{p := 1}](\bot \to \varphi)$, and from this by applying the K-schema $[\boldsymbol{p := 1}]\bot \to [\boldsymbol{p := 1}]\varphi$. In a similar way, we arrive at

$[\boldsymbol{p := 0}]\bot \rightarrow [\boldsymbol{p := 0}]\varphi$. In this way, we can strengthen the substitution schemas for atomic assignment modalities, as follows:

$$[\boldsymbol{p := 1}]\varphi \leftrightarrow [\boldsymbol{p := 1}]\bot \vee [\top/p]\varphi$$

$$[\boldsymbol{p := 0}]\varphi \leftrightarrow [\boldsymbol{p := 0}]\bot \vee [\bot/p]\varphi$$

This gives us a set of equivalences, but note that due to the non-reducibility of $[\boldsymbol{p := 1}]\bot$, our earlier translation procedure for reducing the logic to a purely propositional format breaks down. We now have a genuine dynamic logic, which can be analyzed by standard modal techniques. This logic is complete, for all its axioms have Sahlqvist form, so it has the finite model property, by the canonical model construction plus filtration, hence it is decidable.

Again, one might wish to analyze reasoning over subset models directly in the command language. It is clear that the axioms $\Longrightarrow p := B$ will have to go, as they are no longer sound under the new interpretation. I conjecture that the resulting system is complete for imperative reasoning over subset models.

5. Application: Database Updating Under Constraints

We can apply this to database updating under constraints, by letting a subset model M be given via a constraint C (a contingent formula of propositional logic). C gives the model M via

$$M := \{w \in \{0,1\}^P \mid w \models C\}.$$

The analysis remains much as before, only the definition of μ has to take C into account. Also, the canonical forms of commands become slightly more involved.

6. Further Questions about Action Without Repercussion

Define the command $p := \neg p$ as $p?; p := 0 \cup \neg p?; p := 1$. The difference between a command like $p := 1$ and a command like $p := \neg p$ is that the latter can be undone (by performing it again), because it remembers its previous state, so to speak. $p := \neg p$ is reversed, simply by performing it again. This continues to hold in the logic of action without repercussion over subset models, for we have:

$$\models p \wedge \langle \boldsymbol{p := 0} \rangle \top \rightarrow \langle \boldsymbol{p := 0; p := 1} \rangle \top$$

$$\models \neg p \wedge \langle \boldsymbol{p := 1} \rangle \top \rightarrow \langle \boldsymbol{p := 1; p := 0} \rangle \top$$

Note, however, that the reversibility operator $\breve{}$ cannot be defined in this language, for the simple reason that a command like $p := 1$ does not remember its previous state.

Reversibility cannot even be defined for the subset of those commands built from $p := \neg p$ by means of $;, \cup$. After performing $p := \neg p \cup q := \neg q$ we do not know which of the two we should undo.

We can undo commands in a weaker sense, as follows. Define the *weak reversal* $\breve{}$ of any command π by means of:

$$\begin{array}{lcl}(p := 1)\breve{} & := & \top? \cup p := 0 \\ (p := 0)\breve{} & := & \top? \cup p := 1 \\ (\pi_1; \pi_2)\breve{} & := & \pi_2\breve{}; \pi_1\breve{} \\ (\pi_1 \cup \pi_2)\breve{} & := & \pi_1\breve{} \cup \pi_2\breve{} \\ (\varphi?)\breve{} & := & \top?\end{array}$$

Then we can show that $[\![\pi; \pi\breve{}]\!] \neq \emptyset$. In other words, in this simple constellation an action that is regretted can always be repaired (indeterministically), quite unlike the situation in the real world of spilt milk and broken china.

Another reflection of the simplicity of the set-up is the fact that the order in which basic actions are performed does not matter. We have: $\langle \boldsymbol{p := B_1; q := B_2} \rangle \varphi \leftrightarrow \langle \boldsymbol{q := B_2; p := B_1} \rangle \varphi$, provided that p and q are different. For complex actions, this does not hold, of course. E.g., using the abbreviations mentioned above, $p := q; q := p$ will have a different effect from $q := p; p := q$.

Next, one might ask whether adding Kleene * increases the expressive power of this language. To see that the answer to this question is 'no', observe that an arbitrary command π^* only affects a finite number of atomic propositions. This entails that π^* can always be given in canonical if-then-else form, i.e., can be rephrased without use of *. Finding an efficient method for translating π^* commands into *-free form is another matter. A further question one might also ask is whether there is an efficient method for translating $\mu\pi$ commands into μ-free form.

Note that the schema $\langle \boldsymbol{\pi_1} \rangle \varphi \leftrightarrow \langle \boldsymbol{\pi_2} \rangle \varphi$ is valid iff $[\![\pi_1]\!] = [\![\pi_2]\!]$. One might consider adding a construction to the language to express this directly (i.e., in a single formula). This leads to a next question: Does adding formulas of the form $\pi_1 \equiv \pi_2$, with semantic clause $w \models \pi_1 \equiv \pi_2$ iff for all v: $w, v \models \pi_1$ iff $w, v \models \pi_2$, increase the expressive power of the language? Again, the answer is 'no', and the reason has again to do with the fact that a program π only affects the value of the finitely many proposition letters occurring in it, leaving everything else unchanged. Equivalence of programs π_1 and π_2 depends only on the values of the proposition letters which occur in either of them. Let this set be $\{p_1, \ldots, p_n\}$. Then $\pi_1 \equiv \pi_2$ is equivalent to the following conjunction:

$$\begin{array}{ll}
\langle \boldsymbol{\pi_1} \rangle (p_1 \wedge p_2 \wedge \cdots \wedge p_n) \leftrightarrow \langle \boldsymbol{\pi_2} \rangle (p_1 \wedge p_2 \wedge \cdots \wedge p_n) & \wedge \\
\langle \boldsymbol{\pi_1} \rangle (\neg p_1 \wedge p_2 \wedge \cdots \wedge p_n) \leftrightarrow \langle \boldsymbol{\pi_2} \rangle (\neg p_1 \wedge p_2 \wedge \cdots \wedge p_n) & \wedge \\
\vdots & \wedge \\
\langle \boldsymbol{\pi_1} \rangle (\neg p_1 \wedge \neg p_2 \wedge \cdots \wedge \neg p_n) \leftrightarrow \langle \boldsymbol{\pi_2} \rangle (\neg p_1 \wedge \neg p_2 \wedge \cdots \wedge \neg p_n). &
\end{array}$$

Finally we can ask questions about sensible extensions of the framework. For example, which extension of the framework would be needed to make sense of a command negation operation, for expressing that certain things should *not* be done?

To see that this is an interesting question, note that *prima facie* it seems that the concept of giving an order to *not do* certain things presupposes some kind of inherent change in the world. ‘Do not step on my foot’ typically is shouted at someone who is about to engage in an action that the speaker intends to avoid. Of course, the meaning is not: ‘you can do anything, provided it is not stepping on my foot’, but rather something like: ‘You should minimally modify the course of your impending action, in order to ensure that my foot does not suffer from it.’ Thus, a reasonable treatment of negated action seems to presuppose the concepts of inherent changes in a situation and minimal modifications of inherent change in a situation.

To wind up the discussion of this simple system, let us note that as a treatment of the declarative versus imperative distinction, the logic presented leaves quite a few things to be desired. The three main shortcomings are the following:

- The internal structure of the facts that constitute the world is disregarded.
- The distinction between giving an order (issuing a request) and obeying that order (complying with the request) is disregarded.
- The coherence of the facts that constitute the world, and the circumstance that actions do have repercussions is disregarded.

The first shortcoming is inherent to a propositional treatment, and can be overcome by switching to level where the predicate argument structure of the atoms gets analyzed. The second shortcoming can be overcome in a setting where we distinguish between actual change in the world and intended changes of the world, where the addressee is made responsible for the change. See the work of Noel Belnap c.s. (1992) for this refinement, which is beyond the scope of the present paper. The third shortcoming is addressed in the next section.

7. The Logic of Action With Repercussion

What does it mean that a basic action $p := B$ has a repercussion or side effect? Simply, that I cannot perform that action without affecting the value of other propositions besides p. In general, there are two main causes for this:

1. the inherent causal coherence of the world,

2. my lack of skill.

To distinguish between (1) and (2), some further refinements are necessary that we will not yet make (but see Section 9 below).

To model the concept of 'coherence of the world/lack of skill of the agent', assume models of the form $M = \langle W, R^{\oplus p}, R^{\ominus p}, \ldots \rangle$, where $W \subseteq \{0,1\}^P$, every accessibility relation R is a subset of W^2, and the accessibility relations satisfying the following:

– if $wR^{\oplus p}v$ then $v(p) = 1$,

– if $wR^{\ominus p}v$ then $v(p) = 0$.

Note that these are still rather special Kripke models, because of the circumstance that different worlds cannot have the same valuation. Call such models 'Kripke valuation models'.

If we drop the constraint $W \subseteq \{0,1\}^P$ on W, and instead introduce a valuation V, and replace the constraint on accessibility by:

– if $wR^{\oplus p}v$ then $V_v(p) = 1$,

– if $wR^{\ominus p}v$ then $V_v(p) = 0$.

we get 'Kripke assignment models'.

Note that from a point of view of modal logic the constraints on $R^{\oplus p}, R^{\ominus p}$ put conditions on the class of valuations that fit a frame for this logic. Technically, this constitutes a move towards a semantics in terms of generalized modal frames rather than modal frames.

The language for this is the same as before, with one small addition:

$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid [\boldsymbol{\pi}]\varphi.$

$\pi ::= p := 1 \mid p := 0 \mid \pi;\pi \mid \pi \cup \pi \mid \varphi? \mid \mu\pi.$

The interpretation of formulas and procedures remains the same as before, except for the facts that now we use the accessibility relations to model the change transitions and a notion of 'dissimilarity' between worlds to model the effect of the minimality operator μ. We use the following dissimilarity measure between worlds:

$$\mathrm{Dissim}(w, v) = |\{p \in P \mid w(p) \neq v(p)\}|.$$

Note that we have: $\mathrm{Dissim}(w, w) = 0$ (a world has no dissimilarities with itself), $\mathrm{Dissim}(w, v) = \mathrm{Dissim}(v, w)$, and $\mathrm{Dissim}(w, v) \leq \mathrm{Dissim}(w, u) + \mathrm{Dissim}(u, v)$.

The semantic clauses that differ from those in the previous system are:

- $M, w, v \models p := 1$ iff $wR^{\oplus p}v$.
- $M, w, v \models p := 0$ iff $wR^{\ominus p}v$.
- $M, w, v \models \mu\pi$ iff $M, w, v \models \pi$ and there is no t with $M, w, t \models \pi$ and $\mathrm{Dissim}(w, t) < \mathrm{Dissim}(w, v)$.

The intuition behind the semantic clauses for changing facts is that changing a fact might be impossible in a situation (the case where in the world under consideration there is no accessible world for that change), or it might bring about changes in other facts too (the case where an accessible world does not only differ from the current world in the fact that was to be changed but in other facts as well). The minimality operator plays an essential role in this set-up, for the command to make fact p true is most often supposed to be interpreted as 'Make p true while leaving the rest of the world undisturbed, as far as possible'. In our system, this is rendered as $\mu p := 1$.

The idea of repercussions of actions is brought out very clearly in terms of the μ operator. In a situation w where p is not true we cannot do p without repercussion on other things in case for all v with $M, w, v \models \neg p?; \mu p := 1$ it holds that $\mathrm{Dissim}(w, v) > 1$.

Note that in the present system, unlike the system of the previous section, it does not in general hold that $\mathcal{M} \models \langle \boldsymbol{p := 1} \rangle \top \rightarrow \langle \boldsymbol{p := 1; p := 0} \rangle \top$, for $p := 1$ might be an action which, when done in world w, cannot always be undone. This is the case if there is a v with $wR^{\oplus p}v$, but for no such v: $vR^{\ominus p}w$. In other words, it may be that there are things that cannot be undone. It follows immediately that a converse operation is not definable in this system (not even a weak one).

In a sense, the present system also provides a minimalist account of causation. If the action $p := 1$ always has as a result that we end up in a world where q holds, then, in a minimalist sense, p causes q. Of course, there is much more to be said about causation in connection with the logic of action (see, e.g., Shoham (1988)).

Some obvious axioms for this system are:

1. All instantiations of axiom schemas of propositional logic.
2. $[\boldsymbol{\pi}](\varphi \to \psi) \to ([\boldsymbol{\pi}]\varphi \to [\boldsymbol{\pi}]\psi)$
3. $[\boldsymbol{p := 1}]p$,
   $[\boldsymbol{p := 0}]\neg p$.
4. $[\boldsymbol{\pi_1; \pi_2}]\varphi \leftrightarrow [\boldsymbol{\pi_1}][\boldsymbol{\pi_2}]\varphi$.
5. $[\boldsymbol{\pi_1 \cup \pi_2}]\varphi \leftrightarrow [\boldsymbol{\pi_1}]\varphi \wedge [\boldsymbol{\pi_2}]\varphi$.
6. $[\boldsymbol{\varphi?}]\psi \leftrightarrow (\varphi \to \psi)$.
7. $[\boldsymbol{\pi}]\varphi \to [\boldsymbol{\mu\pi}]\varphi$.
8. $\langle\boldsymbol{\mu\pi}\rangle\varphi \to [\boldsymbol{\mu\pi}]\varphi$,
   provided φ is purely propositional.
9. $[\boldsymbol{\mu(\pi_1; \pi_2)}]\varphi \leftrightarrow [\boldsymbol{\mu\pi_1; \mu\pi_2}]\varphi$.
10. $[\boldsymbol{\mu(\pi_1 \cup \pi_2)}]\varphi \leftrightarrow [\boldsymbol{\mu\pi_1 \cup \mu\pi_2}]\varphi$.
11. $[\boldsymbol{\mu\varphi?}]\psi \leftrightarrow [\boldsymbol{\varphi?}]\psi$.

Rules: modus ponens and necessitation for any π.

Note that the axiom scheme of independent change for atomic assignment variables has gone. The axioms and rules given above are obviously sound for the intended interpretation. I leave the question open whether additional axiom schemas are needed to get a complete calculus. I conjecture that this logic is decidable.

The constraints 'if $wR^{\oplus p}v$ then $v(p) = 1$' and 'if $wR^{\ominus p}v$ then $v(p) = 0$' can be seen as constraints on the way the accessibility relations are connected. It follows from them, for instance, that $\text{rng}\,(R^{\oplus p}) \cap \text{rng}\,(R^{\ominus p}) = \emptyset$, for every $p \in P$.

8. Conditions on Accessibility

We should now ask ourselves what are plausible further constraints on the accessibility relations and the way they are connected?

Here is an example of such a constraint: for every $p \in P$, $R^{\oplus p} \cap \{w \in W \mid w(p) = 1\}$ and $R^{\ominus p} \cap \{w \in W \mid w(p) = o\}$ are reflexive.

The following axiom schemata impose this constraint:

– $(\varphi \wedge p) \to \langle\boldsymbol{p := 1}\rangle\varphi$
– $(\varphi \wedge \neg p) \to \langle\boldsymbol{p := 0}\rangle\varphi$.

Another example of a constraint: $R^{\oplus p} \circ R^{\oplus q} \subseteq R^{\oplus q}$, expressing that if you can reach a situation where q is true by first doing $p := 1$ and then doing $q := 1$, then the same effect can also be had immediately by doing $q := 1$.

Further questions about the plausibility of this and similar constraints might be asked. For instance, is it reasonable to impose the constraint that the $R^{\odot p}$ be transitive?

We have seen above that in action logic without repercussion basic flip actions (actions of the form $p := \neg p$) are reversible: just perform them again. If actions can have repercussions, then this no longer holds: there is no guarantee that for all v we have $M, v, v, \models p := \neg p; p := \neg p$. This shows that we are moving closer to the real world of spilt milk and broken china. Full reversibility would destroy all side effects, of course, but one might want to consider the weaker constraint that the basic flip relation be symmetric. This is expressed by:

$$\varphi \rightarrow [\boldsymbol{p := \neg p}]\langle \boldsymbol{p := \neg p}\rangle\varphi.$$

This expresses that it is always *possible* that things get completely restored, although there is no *guarantee* that they will.

A very strong constraint is that the *order* in which basic actions are performed should not matter. It may still be plausible for specific action pairs $p := B_1; q := B_2$, though. This constraint is expressed by:

$$\langle \boldsymbol{p := B_1}\rangle\langle \boldsymbol{q := B_2}\rangle\varphi \leftrightarrow \langle \boldsymbol{q := B_2}\rangle\langle \boldsymbol{p := B_1}\rangle.$$

It is clear that here is a lot of work to be done: explore the list of plausible constraints, and see if they can be axiomatized.

For the logic of action with repercussion, it is clear that adding Kleene * increases the expressive power of the language. For instance $\langle \boldsymbol{(p := 1; q := 1)^*}\rangle\varphi$ expresses that there is a world v with $w(R^{\oplus p} \circ R^{\oplus q})^* v$ with $v \models \varphi$ (where w is the current world). This is not expressible without *.

Again, questions galore: Give an axiomatization of the minimal system with Kleene star added to the language. Is the minimal logic of change with repercussion and Kleene star decidable? Does adding formulas of the form $\pi \equiv \pi$, with semantic clause $w \models \pi_1 \equiv \pi_2$ iff for all v: $w, v \models \pi_1$ iff $w, v \models \pi_2$, increase the expressive power of the language?

9. Conclusion

By way of conclusion, I briefly comment on some possible application directions for the framework that I have presented above. In the first place, let us take a quick look at what it means that an agent is *able* to perform a basic action $p := B$ in a state w? In a single-agent framework simply this: $\exists v : wR^{Bp}v$. In a multi-agent framework we need basic actions of the form $\mathrm{do}_i p := B$, indicating that the action is performed by agent i, and an ability function

$$\mathit{ABILITY}\colon \text{ agents } \times \text{ propositions } \times \{0,1\} \times W \to \{0,1\}$$

satisfying

$$\text{if } \mathit{ABILITY}(i,p,B,w) = 1 \text{ then } \{v \mid wR^{Bp}v\} \neq \emptyset.$$

This expresses that i can only do things (in a given situation) that can be done at all in that situation. An application for ability models would be multi-agent database management with updating permits.

Next, what does it mean to be able to perform a basic action *with skill*? In our framework: to be able to perform it without unnecessary side effects. This concept is modeled in a skill model, which is a natural generalization of an ability model. In a skill model, differences in skill between agents are accounted for by means of a function

$$\mathit{SKILL}\colon \text{ agents } \times \text{ propositions } \times \{0,1\} \times W \to \mathcal{P}W,$$

satisfying:

$$\text{if } \mathit{SKILL}(i,p,B,w) \neq \emptyset \text{ then } \mathit{SKILL}(i,p,B,w) \supseteq \{v \mid wR^{Bp}v\}.$$

This expresses that if person i is able to perform $p := B$ in situation w at all, then i will perform the action with at least as many (possible) side effects as is inherent in the nature of things. We can say now that person i has maximum skill at action $p := B$ in situation w if $\mathit{SKILL}(i,p,B,w) = \{v \in W \mid wR^{Bp}v\}$. In a multi-agent setting, every basic action comes with a specification of the agent performing it, so basic actions have the form $\mathrm{do}_i p := B$, with interpretation: $M, w, v \models \mathrm{do}_i p := B$ iff $v \in \mathit{SKILL}(i,p,B,w)$.

It is clear, then, that various concepts from action logics in Artificial Intelligence already show up in the present very simple setting:

observation of φ by agent i: $\mathrm{do}_i\varphi?$.

ability of i to do A: $\{\langle w,v\rangle \mid M,w,v \models \mathrm{do}_i A\} \neq \emptyset$.

ability of i to do A at w: $\{v \mid M,w,v \models \mathrm{do}_i A\} \neq \emptyset$.

ability of i to achieve φ by doing A: $\{v \mid M, w, v \models \text{do}_i A\} \models \varphi$.

ability versus skill: $\{v \mid M, w, v \models \text{do}_i A\} \neq \emptyset$ versus $\{v \mid M, w, v \models \text{do}_i A\} = \{v \mid M, w, v \models A\}$.

Next, one can turn to the logic of action in a dynamic world. Enrich models with a function

$$PROCESS : W \times T \rightarrow \mathcal{P}W$$

encoding the ways in which the world may run if I do not act (T is the set of time points). Add a new operator $\Box\varphi$, to be interpreted as 'φ is inevitable if I do not act'. In this setting it is possible to study Von Wright's (1983) distinctions:

make p: $\neg p \wedge \Diamond\neg p \wedge \langle A\rangle\top \wedge [A]p$.

keep p: $p \wedge \Diamond\neg p \wedge \langle A\rangle\top \wedge [A]p$.

end p: $p \wedge \Diamond p \wedge \langle A\rangle\top \wedge [A]\neg p$.

prevent p: $\neg p \wedge \Diamond p \wedge \langle A\rangle\top \wedge [A]\neg p$.

A step in a quite different direction would be to look at the internal structure of facts, by developing predicate logical versions of all these logics. But it seems to me that quite a lot of interesting work remains to be done at the propositional level.

Acknowledgements

Thanks are due to Johan van Benthem, Stijn van Dongen, Maarten de Rijke and an anonymous referee for helpful comments on a previous version of the paper, and to the editors of this volume for their persistence in letting it see the light of day.

References

N. Belnap and M. Perloff. The way of the agent. *Studia Logica*, 51:463–484, 1992.

J. van Benthem. Logic and the flow of information. In *Proceedings 9th International Congress of Logic, Methodology and Philosophy of Science. Uppsala 1991*, pages 693–724. Elsevier, Amsterdam, 1993.

J. van Benthem. *Exploring Logical Dynamics.* CSLI & Folli, 1996.

Y. Shoham. *Reasoning about Change.* MIT Press, Cambridge, Mass, 1988.

P. Spruit, R. Wieringa, and J.-J. Meyer. Axiomatization, declarative semantics and operational semantics of passive and active updates in logic databases. *Journal of Logic and Computation*, pages 27–70, 1995.

G.H. von Wright. *Practical Reason.* Blackwell, Oxford, 1983.

Address for Offprints: Jan van Eijck
CWI
PO Box 94079
1090 GB Amsterdam