

Fixed Point Theory for Consistent Rewriting in Logic of Action with Reverse

Susumu Yamasaki

Okayama University, Okayama, Japan

This paper studies an action structure to recursively derive actions by rewriting, where the action is abstract as in process algebra and has its reverse action. A representation method of action denotations with action defaults is given by fixed point theory, where the denotations are owing to coexistence of actions and their reverses. As the next method, the reverse action is thought of as multiplicative inverse in a semiring, which can be formed by algebraic compositions of action structures to primarily express sequences of actions with the action denotations. In both ways, consistency to classify each action and its reverse into different sorts is respected.

1 Introduction

Hennessy-Milner Logic, which is a framework related to process algebra ([7, 10, 12]), may conceive dynamic changes caused by actions ([6]) and be related to mobile ambients ([3, 11]) with reference to action functions. Actions of agents are captured in strategic reasoning ([8]) and in logical formulas ([14, 18]). Action semantics ([13]) and constraints (as in [2]) may be viewed as containing actions for procedures causing state transitions or computing environmental changes. In [18], a state constraint class of functions (λ -terms) is discussed such that mobility of functions may be modeled. However, it may be just thought of as transferring in a graph, compared with constraint programming. As regards *state* and *change*, problem-solving techniques often contain their concepts ([5]). Trends in weighted automata ([4]) are respected as offering various fundamentals of action sequences (causing state transitions).

With such backgrounds, the author is designing an agent model with both a function module and an action-evocation module to work at designated states for decision processes like adjusting. This motivates a modeling of the structures of action evocations and action sequences causing state-transitions. An abstract mechanism of action-evocations is here considered by the name of an *action structure* as a rewriting system to derive actions and reverse actions, denoting defaults of actions. The structure is constructed by recursive relations among action names so that the rewriting system may recursively derive actions, excluding denoted defaults. From a denotational view, the action and its reverse are categorized into different sorts. From the view of action sequences, the action just followed up by its reverse (and vice versa) may be reduced (rewritten) to the empty (sequence). In this paper, the first denotational view is discussed by fixed point theory in a rewriting system. Based on the denotational views, the second aspect is taken in the context of an reduction of reverses as inverses in semiring.

2 Rewriting in Action Structures

The concept of the state as in formal systems above mentioned is here assumed. An abstract action causing state transitions is supposed, as well. A set of rules to evoke actions is examined, with a relevance to Hennessy-Milner Logic (HML) abstracted from process algebra ([7, 12]). Hennessy-Milner Logic has

an interesting feature even with respect to logic of actions as in [10] and with relation to *knowledge representation* as in [14, 18]. Formulae in HML have the syntax: $\varphi ::= \text{tt} \mid \varphi \wedge \psi \mid \neg\varphi \mid \langle a \rangle \varphi$.

The denotation $\llbracket \varphi \rrbracket$ of a formula φ on a transition system $\mathcal{T} = (S, A, \rightarrow)$ is defined for a set S of states: $\llbracket \text{tt} \rrbracket = S$, $\llbracket \varphi \wedge \psi \rrbracket = \llbracket \varphi \rrbracket \cap \llbracket \psi \rrbracket$, $\llbracket \neg\varphi \rrbracket = S - \llbracket \varphi \rrbracket$, $\llbracket \langle a \rangle \varphi \rrbracket = \{s \in S \mid \exists t \in S: s \xrightarrow{a} t \text{ and } t \in \llbracket \varphi \rrbracket\}$, where an expression $\langle a \rangle$ is concerning an abstract action a in A and “ $\rightarrow$ ” is a state-transition relation.

The set of rules is defined as recursively deriving evocative actions by rewriting an action (name) in a rule with a set of evocative and default actions. The set of rules is in this paper called an *action structure* in that (i) it is a structure to evoke actions and (ii) it may be structurally composed with others to derive action sequences. A default of each action for evocation is supposedly sensitive to the state so that the action structure may contain action defaults in a rule. An evoked action is abstractly assumed to be implementable in this paper. So far notations of action implementation and default are briefly given as below. As regards the organization of an action structure R (as a rule set) over both a set A of actions and a set of action defaults $\langle A \rangle_d = \{\langle a \rangle_d \text{ (default of } a) \mid a \in A\}$, the relation $\succ \subseteq 2^{(A \cup \langle A \rangle_d)} \times (A \cup \langle A \rangle_d)$ may be used. An expression of the *finite* form $X \succ \alpha$ is a rule such that an action or action default α may be implementable if all the members in X are implementable. An action default is supposed as implementable iff the evocation for the action is a default. That is, $\langle a \rangle_d$ can be regarded as implementable iff the evocation of an action a is a default. Such a rule can be interpreted as rewriting of an action or an action default α to the set X . In this paper, we assume that at most one X precedes α for each action or default α . That is, α may be rewritten in a deterministic way or not. This paper deals with the case that the set A of actions may possibly contain reverse actions. By the expression a^{-1} , we mean a reverse (action) to each action a . While we assume $(a^{-1})^{-1} = a$ so that each action may be at the same time a reverse of its reverse action, a categorization is, by renaming of actions, made in what follows such that an originally taken action as the positive may not contain the notation “ -1 ” and its reverse action as the negative may contain only one notation “ -1 ”.

We may recursively have *implementability* predicate $\text{imple}_R(-)$, and *default* predicate $\text{default}_R(-)$, with reference to a rule set R , where: for a rule $X \succ \alpha$,

- (a) if $\forall a \in X. \text{imple}_R(a)$, and $\forall \langle b \rangle_d \in X. \text{default}_R(b)$, then $\text{imple}_R(\alpha)$, and
- (b) if $\exists a \in X. \text{default}_R(a)$, or $\exists \langle b \rangle_d \in X. \text{imple}_R(b)$, then $\text{default}_R(\alpha)$.

$\text{imple}_R(\langle b \rangle_d)$ is interpreted as equivalent to $\text{default}_R(b)$. Note that if $X = \emptyset$ in a rule of the form $X \succ \alpha$, then $\text{imple}_R(\alpha)$. Assume that if there is no rule of the form $X \succ a$, then $\text{default}_R(a)$. We may have a fixed point theory, when we assume for consistent derivation by rewriting rules that the denotation of an action should be distinguished from the one of its reverse, with the view of implementation.

Consistency: This paper takes consistency view of whether the action structure R over A and $\langle A \rangle_d$ is constructed such that the predicates $\text{imple}_R(a)$ and $\text{imple}_R(a^{-1})$ cannot be compatible for each action a , where the sets $\{e \in A \mid \text{imple}_R(e)\}$ and $\{b \in A \mid \text{default}_R(b)\}$ are disjoint with each other.

3 Fixed Point Method for Rewritten Derivations

This section describes the denotations given by an amendment of the action structure R over the set $A \cup \langle A \rangle_d$ (which is finite or countably infinite). To keep any action free from its reverse, the given action structure R is replaced by the action structure

$$R_0 = \{X \cup \{\langle a \rangle_d\} \succ a^{-1} \mid \text{there is a rule } X \succ a^{-1} \text{ in } R \text{ for the reverse action } a^{-1}\} \\ \cup \{X \succ \alpha \mid X \succ \alpha \text{ is in } R \text{ and } \alpha \text{ is not a reverse action of the form } a^{-1}\}.$$

The reason to replace R by R_0 comes from the idea like *exception* for adductive logic programs (R.Kowalski et al., 1990) such that implementability denoted by R_0 may fit constraint rewritings in R .

- (i) The implementation of an action a excludes the implementation of its reverse action a^{-1} , such that the rule regarding the reverse a^{-1} contains $\langle a \rangle_d$ in the rewritten part.
- (ii) The implementation of an action of the form a or an action default can be ruled by the given structure R .

Given an action structure R_0 over the set A of actions and the default set $\langle A \rangle_d$, let $D = A \cup \langle A \rangle_d$. A mapping $T_{R_0} : 2^A \rightarrow 2^D$ is defined such that: $\alpha \in T_{R_0}(B) \Leftrightarrow \exists (X \succ \alpha) \text{ in } R_0. (X \subseteq B)$.

Because the mapping T_{R_0} is monotonic and continuous with the partial order of subset inclusion (as the discussion even in [17]), the least fixed point of T_{R_0} exists and is given by $T_{R_0} \uparrow \omega = \cup_{i \in \omega} T_{R_0}^i(\emptyset)$, where $T_{R_0}^i(\emptyset)$ is a set obtained by i times applying T_{R_0} to the empty set.

For a subset $C \subseteq A$, a rule set $R_0[C]$ is constructed by: $X - \langle C \rangle_d \succ \alpha$ is in $R_0[C] \Leftrightarrow X \succ \alpha$ is in R_0 , where $\langle C \rangle_d = \{ \langle a \rangle_d \mid a \in C \}$. A subset $V_{R_0}(C)$ of A is given to denote the set of action defaults on the assumption of the set C : $b \in V_{R_0}(C) \Leftrightarrow \langle b \rangle_d \in T_{R_0[C]} \uparrow \omega$. A mapping $U_{R_0} : 2^A \rightarrow 2^A$ is defined by: $a \in U_{R_0}(C_0) \Leftrightarrow a \in T_{R_0[C_0]} \uparrow \omega$.

Given a subset C_0 of A , we can, by recursive derivations of the rule set R_0 and by fixed point induction included in U_{R_0} , see the *denotation* properties:

- (1) $a \in U_{R_0}(C_0) \Leftrightarrow \text{imple}_{R_0}(a)$ on condition that $\forall b \in C_0. \text{default}_{R_0}(b)$.
- (2) $a \in U_{R_0}(\overline{U_{R_0}(C_0)}) \Leftrightarrow \text{imple}_{R_0}(a)$ or *not* $\text{default}_{R_0}(a)$ on condition that $\forall b \in C_0. \text{default}_{R_0}(b)$.
- (3) $a \in \overline{U_{R_0}(\overline{U_{R_0}(C_0)})} \Leftrightarrow \text{default}_{R_0}(a)$ on condition that $\forall b \in C_0. \text{default}_{R_0}(b)$.

Note that the above properties are concerned with implementable actions (by the predicate $\text{imple}_{R_0}(-)$) and action defaults (by the predicate $\text{default}_{R_0}(-)$) on the assumption of a default set C_0 .

The implementations of an action and its reverse cannot be compatible, with the conditions like value assignments in 3-valued logic ([15]). On the basis of the above properties, we can, for a given structure R (and its transformation R_0), prepare the equation

$$\overline{U_{R_0}(\overline{U_{R_0}(C \cup V_{R_0}(C))})} = C \cup V_{R_0}(C)$$

to denote the set of both action defaults and possible reverse actions. It is referred to as a *negative set equation* in the sense that it excludes the set of actions to be derived by rewriting from the action structure R_0 . The set R_0 is consistent from the points of classifying each implementable action and its reverse into different sorts. The following definition is concerning the consistency condition that implementability and default of actions cannot be compatible on the basis of a default set C .

Definition 1 The action structure R_0 transformed from a given one R is said to be consistent with respect to the default set $C \cup V_{R_0}(C)$ if $U_{R_0}(C \cup V_{R_0}(C)) \cap (C \cup V_{R_0}(C)) = \emptyset$. By the predicate $\text{consis}(R_0, C \cup V_{R_0}(C))$, we mean that R_0 is consistent with respect to $C \cup V_{R_0}(C)$.

Lemma 2 Assume that the action structure R_0 is consistent with respect to $\emptyset \cup V_{R_0}(\emptyset)$. Then there is a least fixed point of the negative set equation with respect to $\overline{U_{R_0}(\overline{U_{R_0}(-)})}$ such that the action structure is consistent with respect to the fixed point.

Proof. The outline of the proof is viewed, different from the methods and ideas in [1, 9]. Based on the assumption of consistency with respect to $\emptyset \cup V_{R_0}(\emptyset)$, we can see the inductions that: (i) if $\text{consis}(R_0, C_1)$ for $C_1 = C \cup V_{R_0}(C)$, then $\text{consis}(R_0, \overline{U_{R_0}(\overline{U_{R_0}(C_1))})$. (ii) if $\text{consis}(R_0, C_i)$ for any C_i in any chain H , then

$consis(R_0, \cup H)$. The existence of the least fixed point can be guaranteed, since $\overline{U_{R_0}(U_{R_0}(-))}$ is monotonic. With the inductions (i) and (ii) as above, $consis(R_0, C_f)$ for the fixed point C_f . Q.E.D.

We may finally see soundness and completeness of $imple_{R_0}(a)$ and $default_{R_0}(b)$ with respect to the denotations in terms of the fixed point, by means of “denotation properties” as above and with monotonicity of $\overline{U_{R_0}(U_{R_0}(-))}$.

Soundness: Assume that the action structure R_0 is consistent with respect to $\emptyset \cup V_{R_0}(\emptyset)$. Suppose also that the least fixed point of the negative set equation with respect to $\overline{U_{R_0}(U_{R_0}(-))}$ is C_f . Then (1) if $imple_{R_0}(a)$, then a is in $U_{R_0}(C_f)$, and (2) if $default_{R_0}(b)$, then b is in C_f .

Completeness: Assume that the action structure R_0 is consistent with respect to $\emptyset \cup V_{R_0}(\emptyset)$. Suppose also that the least fixed point of the negative set equation with respect to $\overline{U_{R_0}(U_{R_0}(-))}$ is C_f . Then (1) if a is in $U_{R_0}(C_f)$, then $imple_{R_0}(a)$, and (2) if b is in C_f , then $default_{R_0}(b)$.

4 Algebraic Action Structure

This section is a reference to a problem on reduction of reverse actions in the case that the reverse action is abstracted to *multiplicative (concatenated) inverse* in a *semiring*.

As a semiring with multiplicative inverse, we may have an abstraction from the reverse action (role) to a multiplicative inverse, where both of an action and its reverse may occur in a sequence derived by compositions of action structures. An occurrence of non-reverse exclusive-or reverse for each action is considered as consistency for an implementable sequence. This topic is related to the semiring as in [16], so that inverse may possibly be eliminated in sequential form of actions.

With an action structure variable R ranging $\mathcal{R}_A$ (which is the domain consisting of action structures over the set A of actions and the default set $\langle A \rangle_d$), we have an expression r of algebraic compositions recursively defined: $r ::= 0 \mid E \mid R \mid r \oplus r \mid r \circ r \mid \{ r \}$, where (i) 0 is a structure without implementability predicate and with default predicate universally for any action, and E is a structure with implementability predicate only for ε (as a specific action denoting the empty sequence) and without default predicate.

With respect to the predicates $imple_R(a)$ and $default_R(a)$ (which can be expressed by fixed point theory of the previous section) for the action structure R , we recursively have a set $imp(r)$ of implementable action sequences based on implementability predicate:

$$imp(r) = \begin{cases} \emptyset & (r = 0) \\ \{\varepsilon\} & (r = E) \\ \{a \in A \mid imple_R(a)\} & (r = R) \\ imp(r_1) \cup imp(r_2) & (r = r_1 \oplus r_2) \\ imp(r_1) \bullet imp(r_2) \text{ (“}\bullet\text{” is the set concatenation)} & (r = r_1 \circ r_2) \\ imp(r_0)^* \text{ (“}\ast\text{” denotes Kleene star)} & (r = \{ r_0 \}) \end{cases}$$

where the expression ε denotes the empty sequence (of actions) such that $\varepsilon a = a \varepsilon = a$ for each action $a \in A$. We recursively have a set $de(r)$ of defaults, as well:

$$de(r) = \begin{cases} A & (r = 0) \\ \emptyset & (r = E) \\ \{a \in A \mid default_R(a)\} & (r = R) \\ de(r_1) \cap de(r_2) & (r = r_1 \oplus r_2) \\ de(r_1) \cup de(r_2) & (r = r_1 \circ r_2) \\ \emptyset & (r = \{ r_0 \}) \end{cases}$$

Definition 3 Given an expression r , we say that an action sequence w satisfies $\text{consis}(r)$ if w contains no action in $\text{de}(r)$. The set $\text{seq}(r)$ of sequences is defined to be $\{w \in \text{imp}(r) \mid w \text{ satisfies } \text{consis}(r)\}$.

Definition 4 Let Γ be a set of expressions r recursively definable in BNF (as above stated). An equivalence relation $\equiv_\Gamma$ (or $\equiv$) on Γ is defined: $r \equiv r'$ iff $\text{seq}(r) = \text{seq}(r')$.

We may have the following properties to see an idempotent semiring on $\Gamma / \equiv$:

Proposition 5 (i) $(r_1 \oplus r_2) \oplus r_3 \equiv r_1 \oplus (r_2 \oplus r_3)$. (ii) $0 \oplus r \equiv r \oplus 0 \equiv r$. (iii) $r_1 \oplus r_2 \equiv r_2 \oplus r_1$. (iv) $r \oplus r \equiv r$. (v) $(r_1 \circ r_2) \circ r_3 \equiv r_1 \circ (r_2 \circ r_3)$. (vi) $E \circ r \equiv r \circ E \equiv r$. (vii) $r \circ (r_1 \oplus r_2) \equiv (r \circ r_1) \oplus (r \circ r_2)$. (viii) $(r_1 \oplus r_2) \circ r \equiv (r_1 \circ r) \oplus (r_2 \circ r)$. (ix) $0 \circ r \equiv r \circ 0 \equiv 0$.

In a semiring on the quotient set $\Gamma / \equiv$, Kleene star can be defined. As regards a sequence denoted by an expression of Γ , we can see a recursion (possibly related to fixed point theory) to eliminate inverses from a sequence in $\text{seq}(r)$ by the condition that $a \cdot a^{-1} = a^{-1} \cdot a = \varepsilon$, with a removal rule of inverses.

References

- [1] J.J. Alferes, C.V. Damasio & J.M. Pereira (1995): *A logic programming system for nonmonotonic reasoning*. *J.Autom.Reasoning* 14(1), pp. 93–147.
- [2] C. Bertolissi, H. Cirstea & C. Kirchner (2006): *Expressing combinatory reduction systems derivations in the rewriting calculus*. *Higher-Order.Symbolic.Comput.* 19(4), pp. 345–376.
- [3] L. Cardelli & A.D. Gordon (2000): *Mobile ambients*. *Theoret.Comput.Sci.* 240(1), pp. 177–213.
- [4] M. Droste, W. Kuich & H. Vogler (Eds.) (2009): *Handbook of Weighted Automata*. Springer.
- [5] M. Genesereth & N. Nilsson (1987): *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers, Inc.
- [6] L. Giordano, A. Martelli & C. Schwind (2000): *Ramification and causality in a modal action logic*. *J.Log.Comput.* 10(5), pp. 625–662.
- [7] M. Hennessy & R. Milner (1985): *Algebraic laws for nondeterminism and concurrency*. *J.ACM* 32(1), pp. 137–161.
- [8] W. van der Hoek, W. Jamroga & M. Wooldridge (2005): *A logic for strategic reasoning*. 4th AAMAS: *Proceedings*, pp. 157–164.
- [9] K. Inoue & C. Sakama (1998): *Negation as failure in the head*. *J.Log.Program.* 35(1), pp. 39–78.
- [10] A. Kucera & J. Esparza (2003): *A logical viewpoint on process-algebraic quotients*. *J.Log.Comput.* 13(6), pp. 863–880.
- [11] M. Merro & F.Z. Nardelli (2005): *Behavioural theory for mobile ambients*. *J.ACM* 52(6), pp. 961–1023.
- [12] R. Milner (1999): *Communicating and Mobile Systems: The Π -Calculus*. Cambridge University Press.
- [13] P.M. Mosses (1992): *Action Semantics*. Cambridge University Press.
- [14] R. Reiter (2001): *Knowledge in Action*. MIT Press.
- [15] T. Reps, M. Sagiv & R. Wilhelm (2004): *Static program analysis via 3-valued logic*. 16th CAV: *Proceedings*, pp. 15–30.
- [16] T. Reps, S. Schwoon & S. Jha (2005): *Weighted pushdown systems and their application to interprocedural dataflow analysis*. *Sci.Comput.Program.* 58(1-2), pp. 206–263.
- [17] S. Yamasaki (2006): *Logic programming with default, weak and strict negations*. *Theory.Pract.Log.Program.* 6(6), pp. 737–749.
- [18] S. Yamasaki (2010): *A construction of logic-constrained functions with respect to awareness*. *LRBA, MAL-LOW: Proceedings*, pp. 84–98.