Characterizing Recursive Programs Up To Bisimilarity

Paul Blain Levy
University of Birmingham

Abstract

A recursive program is determined, up to bisimilarity, by the operation of the recursion body on arbitrary processes, of which it is a fixpoint. The traditional proof of this fact uses Howe's method, but that does not tell us how the fixpoint is obtained.

In this paper, we show that the fixpoint may be obtained by a least fixpoint procedure iterated through the hierarchy of countable ordinals, using Groote and Vaandrager's notion of nested simulation.

1 Introduction

Recursion is an important programming language feature that provides a fixpoint of an endofunction. But an endofunction may have many fixpoints, so an important question in semantics is to determine which is the one calculated by recursion. For example, for both the may-testing and must-testing preorders, recursion calculates the *least* pre-fixed point. (In the case of must-testing, we must assume the calculus uses erratic rather than ambiguous nondeterminism, see e.g. [Las98].)

What if we work modulo bisimilarity? We provide a characterization of the fixpoint calculated by recursion as follows. First calculate the least pre-fixed point up to similarity. Within this equivalence class, calculate the least pre-fixed point up to 2-nested similarity. Iterate this procedure through all the countable ordinals and it converges on a single point: the "nesting fixpoint". This is the one that recursion calculates.

In Sect. 2 we introduce the general notions of nested simulation and nesting fixpoints. We illustrate them in Sect. 3 with the process calculus CCS.

Notation

- We write ω_1 for the least uncountable ordinal.
- For any sets X,Y,Z and relations $\mathscr{R} \subseteq X \times Y$ and $\mathscr{R}' \subseteq Y \times Z$, we write $\mathscr{R};\mathscr{R}'$ for the composite.
- For any sets X and Y, we write $X \rightharpoonup Y$ for the set of partial functions from X to Y with finite domain.
- For any set X with preorder $\leqslant$, and any $\leqslant \cap \geqslant$-equivalence class U, we write

$$\downarrow^{\leqslant}(U) \stackrel{\text{def}}{=} \{x \in X \mid \exists y \in X.\, x \leqslant y\} = \{x \in X \mid \forall y \in X.\, x \leqslant y\}$$

2 Transition systems and ω_1-nested preorders

We first recall the basic notions of transition systems.

Definition 1. *Let* Act *be a set of* actions*, and let* $\mathscr{S} = (X, \rightarrow)$ *be an* Act-labelled transition system*, i.e. a set* X *together with a relation* $\rightarrow \subseteq X \times \mathsf{Act} \times X$.

1. *For each $x \in X$ and $a \in \mathsf{Act}$, we write*

$$\mathsf{succ}_a(P) \overset{\text{def}}{=} \{Q \in \mathsf{Prog} \mid P \xrightarrow{a} Q\}$$

   If this set is always countable, $\mathscr{S}$ is said to be image-countable.

2. *A relation $\mathscr{R} \subseteq X \times X$ is*
   - *a* simulation *on $\mathscr{S}$ when for all $(x,x') \in \mathscr{R}$ and $a \in \mathsf{Act}$, if $y \in \mathsf{succ}_a(x)$ then there exists $y' \in \mathsf{succ}_a(x')$ such that $(x',y') \in \mathscr{R}$.*
   - *a* bisimulation *on $\mathscr{S}$ when both $\mathscr{R}$ and its converse are simulations.*

3. *The greatest bisimulation is called* bisimilarity. *It is an equivalence relation and written $\simeq$.*

4. *Let $\leqslant$ be a preorder on X. A relation $\mathscr{R} \subseteq X \times X$ is*
   - *a* simulation up to $\leqslant$ on the right *when for all $(x,x') \in \mathscr{R}$ and $a \in \mathsf{Act}$, if $y \in \mathsf{succ}_a(x)$ then there exists $y' \in \mathsf{succ}_a(x')$ such that $(x',y') \in (\mathscr{R};\leqslant)$*
   - *a* simulation up to $\leqslant$ *when for all $(x,x') \in \mathscr{R}$ and $a \in \mathsf{Act}$, if $y \in \mathsf{succ}_a(x)$ then there exists $y' \in \mathsf{succ}_a(x')$ such that $(x',y') \in (\leqslant;\mathscr{R};\leqslant)$*

Definition 2. *[GV92] Let Act be a set, and let $\mathscr{S} = (X, \to)$ be an Act-labelled transition system. For each ordinal α, we shall define a preorder $\lesssim_\alpha$, known as* α-nested similarity, *with the property that any simulation contained in $\gtrsim_\alpha$ is also contained in $\lesssim_\alpha$. We define $\lesssim_\alpha$ to be*

($\alpha = \beta + 1$) *the greatest simulation contained in $\lesssim_\beta$, or equivalently the greatest simulation contained in $\lesssim_\beta \cap \gtrsim_\beta$*

(α a limit ordinal) *the intersection of $\lesssim_\beta$ over all $\beta < \alpha$.*

Lemma 1. *[GV92] Let Act be a set, and let $(X, \to)$ be an Act-labelled transition system.*

1. *Let β be an ordinal. If $\mathscr{R}$ is a simulation up to $\lesssim_{\beta+1}$ contained in $\gtrsim_\beta$, then $\mathscr{R}$ is contained in $\lesssim_{\beta+1}$.*
2. *$\lesssim_\alpha$ contains bisimilarity, for each ordinal α.*
3. *If $(X, \to)$ is image-countable, then $\lesssim_{\omega_1}$ is bisimilarity.*

We are thus led to the following abstract notion.

Definition 3. *Let X be a set. An* ω_1-nested preorder *on X is a sequence of preorders $(\leqslant_\alpha)_{\alpha \leqslant \omega_1}$ such that*

- $(\leqslant_{\alpha+1}) \subseteq (\leqslant_\alpha) \cap (\geqslant_\alpha)$*, for each $\alpha < \omega_1$*
- $(\leqslant_\gamma) = \bigcap_{\alpha<\gamma}(\leqslant_\alpha)$*, for each limit ordinal $\gamma \leqslant \omega_1$*

It follows that

- $(\leqslant_0)$ is the indiscrete relation
- $\alpha \leqslant \beta \leqslant \omega_1$ implies $(\leqslant_\beta) \subseteq (\leqslant_\alpha)$
- $\alpha < \beta \leqslant \omega_1$ implies $(\leqslant_\beta) \subseteq (\geqslant_\alpha)$

- $(\leqslant_\alpha)$ is an equivalence relation, for each limit ordinal $\alpha \leqslant \omega_1$.

In particular, $\leqslant_{\omega_1}$ is an equivalence relation, which we write $\equiv$.

Definition 4. *Let* Act *be a set, and let* $\mathscr{S} = (X, \rightarrow)$ *be an* Act*-labelled transition system. We write* $A(\mathscr{S})$ *for the* ω_1*-nested preordered set* $(X, (\lesssim_\alpha)_{\alpha \leqslant \omega_1})$.

Definition 5. *Let* $A = (X, (\leqslant_\alpha)_{\alpha \leqslant \omega_1}$ *and* $B = (Y, (\leqslant_\alpha))_{\alpha \leqslant \omega_1}$ *be* ω_1*-nested preordered sets. A* monotone function $A \xrightarrow{f} B$ *is a function* $X \xrightarrow{f} Y$ *such that, for every* $\alpha \leqslant \omega_1$ *(or equivalently: every successor ordinal* $\alpha < \omega_1$*), if* $x \sqsubseteq_A^\alpha x'$ *then* $f(x) \sqsubseteq_B^\alpha f(x')$.

Now we come to our key definition.

Definition 6. *Let* $A = (X, (\leqslant_\alpha)_{\alpha \leqslant \omega_1}$ *be an* ω_1*-nested preordered set, and let* f *be a monotone endofunction on* A*. We shall define a decreasing sequence of subsets* $(U_\alpha^f)_{\alpha \leqslant \omega_1}$ *of* X *such that* U_α^f *either is empty or satisfies the following conditions:*

- U_α^f *is an equivalence class of* $\leqslant_\alpha \cap \geqslant_\alpha$
- f *restricts to an endofunction on* U_α^f *and hence on* $\downarrow^{\leqslant_\alpha} (U_\alpha^f)$
- *if* $x \in \downarrow^{\leqslant_\alpha} (U_\alpha^f)$ *and* $f(x) \leqslant_\alpha x$ *then* $x \in U_\alpha^f$.

We define U_α^f *to be*

$(\alpha = \beta + 1)$ *the set of* $\leqslant \alpha$*-least elements of*

$$\{x \in U_\beta^f \mid f(x) \leqslant_\alpha x\} = \{x \in \downarrow^{\leqslant_\beta} (U_\beta^f) \mid f(x) \leqslant_\alpha x\}$$

(α a limit ordinal) *the intersection of* U_β^f *over all* $\beta < \alpha$.

The elements of $U_{\omega_1}^f$ *are called* nesting fixpoints *of* f.

Note that nesting fixpoints are fixpoints up to $\equiv$ and unique up to $\equiv$. But some monotone endofunctions f do not have a nesting fixpoint—i.e. $U_{\omega_1}^f$ is empty.

3 CCS and Bisimilarity

Our thesis is that a recursive program in a transition system $\mathscr{S}$ is is a nesting fixpoint of the monotone endofunction on $A(\mathscr{S})$ given by the recursion body. To illustrate this, we consider the calculus CCS [Mil89], over a fixed set Act of actions.

As CCS is untyped, a *context* Γ is merely a list of distinct identifers. The syntax is given inductively by the rules in Fig. 1. We write Prog for the set of *programs*, i.e. closed terms, which forms an Act-labelled transition system with transition relation $\rightarrow$ defined inductively by the rules in Fig. 2. This system is easily shown to be image-countable, and we call it CCS.

Our version of CCS includes parallel composition of any countable arity I, with synchronization described by a relation V saying when finitely many actions performed by the constituent processes may cause an action in the combined process. In [Mil89], Act is given by a disjoint union

$$\{a \mid a \in \Sigma\} \ \cup \ \{\overline{a} \mid a \in \Sigma\} \ \cup \ \{\tau\}$$

where Σ is a set of *synchronization actions*. The parallel composition, hiding and renaming operators provided there are subsumed by our parallel composition as follows.

$$\frac{\Gamma \vdash P}{\Gamma \vdash a.P}\ a \in \mathsf{Act} \qquad \frac{\Gamma \vdash P_i\ \ (\forall i \in I)}{\Gamma \vdash \sum_{i \in I} P_i}\ I \text{ countable} \qquad \frac{\Gamma, \mathtt{x} \vdash P}{\Gamma \vdash \mathtt{rec}\ \mathtt{x}.\ P}$$

$$\frac{}{\Gamma \vdash \mathtt{x}}\ \mathtt{x} \in \Gamma \qquad \frac{\Gamma \vdash P_i\ \ (\forall i \in I)}{\Gamma \vdash \|^{V}_{i \in I}\ P_i}\ I \text{ countable}, V \subseteq (I \rightharpoonup_{\mathsf{fin}} \mathsf{Act}) \times \mathsf{Act}$$

Figure 1: Syntax of CCS

$$\frac{}{a.P \xrightarrow{a} P} \qquad \frac{P_{\hat{\imath}} \xrightarrow{a} Q}{\sum_{i \in I} P_i \xrightarrow{a} Q}\ \hat{\imath} \in I$$

$$\frac{P[\mathtt{rec}\ \mathtt{x}.\ P/\mathtt{x}] \xrightarrow{a} Q}{\mathtt{rec}\ \mathtt{x}.\ P \xrightarrow{a} Q} \qquad \frac{P_i \xrightarrow{b(i)} Q_i\ \ (\forall i \in \mathsf{dom}\ b)}{\|^{V}_{i \in I}\ P_i \xrightarrow{a} \|^{V}_{i \in I} \begin{cases} Q_i & (\text{if } i \in \mathsf{dom}\ b) \\ P_i & (\text{otherwise}) \end{cases}}\ (b, a) \in V$$

Figure 2: Operational Semantics (Transitions) of CCS

- We express $P \mid Q$ as $\|^{V}\ \{0 \mapsto P, 1 \mapsto Q\}$, with V given by

$$\{(\{0 \mapsto a, 1 \mapsto \overline{a}\}, \tau) \mid a \in \Sigma\} \ \cup\ \{(\{0 \mapsto \overline{a}, 1 \mapsto a\}, \tau) \mid a \in \Sigma\}$$
$$\cup\ \{(\{0 \mapsto a\}, a) \mid a \in \mathsf{Act}\} \ \cup\ \{(\{1 \mapsto a\}, a) \mid a \in \mathsf{Act}\}$$

- Let $f : \Sigma \to \Sigma$ be a function. We express $P[f]$ as $\|^{V}\ \{0 \mapsto P\}$, with V given by

$$\{(\{0 \mapsto a\}, f(a)) \mid a \in \Sigma\} \ \cup\ \{(\{0 \mapsto \overline{a}\}, \overline{f(a)}) \mid a \in \Sigma\} \ \cup\ \{(0 \mapsto \tau, \tau)\}$$

- Let $L \subseteq \Sigma$ be a subset. We express $P \backslash L$ as $\|^{V}\ \{0 \mapsto P\}$, with V given by

$$\{(\{0 \mapsto a\}, a) \mid a \in \Sigma \backslash L\} \ \cup\ \{(\{0 \mapsto \overline{a}\}, \overline{a}) \mid a \in \Sigma \backslash L\} \ \cup\ \{(0 \mapsto \tau, \tau)\}$$

The "synchronization algebras" of [WN95] are likewise expressible.

As explained in [Mil89], we could also incorporate into the language countably mutual recursion. We have not done so, but our results would go through without difficulty.

The following operations on programs are called the *basic operations*:

$$\begin{array}{rcll} P & \mapsto & a.P & \text{for any } a \in \mathsf{Act} \\ (P_i)_{i \in I} & \mapsto & \sum_{i \in I} P_i & \text{for any countable } I \\ (P_i)_{i \in I} & \mapsto & \|^{V}_{i \in I}\ P_i & \text{for any countable } I \text{ and } V \subseteq (I \rightharpoonup_{\mathsf{fin}} \mathsf{Act}) \times \mathsf{Act} \end{array}$$

Proposition 1. *The basic operations preserve α-nested similarity, for every ordinal α, and hence preserves bisimilarity.*

Proof. Straightforward induction on α. Preservation of bisimilarity may also be proved directly. □

Lemma 2. *Let $\mathscr{R}$ be a simulation on CCS. Then the relation*

$$\{(M[P/\mathtt{x}], M[P'/\mathtt{x}]) \mid (P, P') \in \mathscr{R}, \mathtt{x} \vdash M\}$$

is also a simulation.

Proof. We want to show that if $M[P/\mathtt{x}] \xrightarrow{a} Q$, then for all P' such that $(P,P') \in \mathscr{R}$ we have $(R,R') \in \mathscr{R}$ and $\mathtt{x} \vdash N$ such that $Q = N[R/\mathtt{x}]$ and $M[P'/\mathtt{x}] \xrightarrow{a} N[R'/\mathtt{x}]$. We proceed by induction on $\rightarrow$. We omit the details. □

Proposition 2. *(Definable functions are monotone) Let $\mathtt{x} \vdash M$ be a term. Then the endofunction on* Prog

$$P \mapsto M[P/\mathtt{x}]$$

preserves α-similarity, for every ordinal α, and hence preserves bisimilarity.

Proof. By induction on α using Lemma 2. Preservation of bisimilarity also follows directly from Lemma 2. □

Lemma 3. *Let $\leqslant$ be a preorder on* Prog *that is a simulation and preserved by the basic operations. Let $\mathtt{x} \vdash M$ and $P \in$* Prog *be such that $M[P/\mathtt{x}] \leqslant P$. Then the relation*

$$\{(N[\mathtt{rec}\,\mathtt{x}.\,M/\mathtt{y}], N[P/\mathtt{y}]) \mid \mathtt{y} \vdash N\}$$

is a simulation up to $\leqslant$ on the right.

Proof. We want to show that if $N[\mathtt{rec}\ \mathtt{x}.\ M/\mathtt{y}] \xrightarrow{a} Q$ then there exists $\mathtt{y} \vdash R$ and $Q' \in \mathsf{Prog}$ such that $Q = R[\mathtt{rec}\,\mathtt{x}.\,M/\mathtt{y}]$ and $N[P/\mathtt{y}] \xrightarrow{a} Q'$ and $R[N/\mathtt{y}] \leqslant Q'$. We proceed by induction on $\rightarrow$.

- Suppose that $N = \mathtt{y}$. Then $M[\mathtt{rec}\ \mathtt{x}.\ M/\mathtt{x}] \xrightarrow{a} Q$ and applying the inductive hypothesis gives $\mathtt{y} \vdash R$ and $Q' \in \mathsf{Prog}$ such that $Q = R[\mathtt{rec}\ \mathtt{x}.\ M/\mathtt{y}]$ and $M[P/\mathtt{x}] \xrightarrow{a} Q'$ and $R[N/\mathtt{y}] \leqslant Q'$. Since $M[P/\mathtt{x}] \leqslant P$ and $\leqslant$ is a simulation we have $P \xrightarrow{a} Q''$ and $Q' \leqslant Q''$, giving $R[N/\mathtt{y}] \leqslant Q''$.
- The other cases are trivial.

□

Proposition 3. *Let $\mathtt{x} \vdash M$ be a term. Then* $\mathtt{rec}\ \mathtt{x}.\ M$ *is a nesting fixpoint of the monotone endofunction $f : P \mapsto M[P/\mathtt{x}]$ on $A(CCS)$.*

Proof. We have to show that $\mathtt{rec}\ \mathtt{x}.\ M$ is in U^f_α for each $\alpha \leqslant \omega_1$. The case where α is a limit ordinal is trivial, so suppose $\alpha = \beta + 1$. Since $\mathtt{rec}\ \ \mathtt{x}.M \in U^f_\beta$ we have

$$\downarrow^{\leqslant_\beta} (U^f_\beta) = \{P \in \mathsf{Prog} \mid P \lesssim_\beta \mathtt{rec}\ \mathtt{x}.\ M\}$$

We need to show that $\mathtt{rec}\ \mathtt{x}.\ M$ is an $\lesssim_{\beta+1}$-least element of

$$\{P \in \downarrow^{\leqslant_\beta} (U^f_\beta) \mid f(P) \lesssim_\alpha P\} = \{P \in \mathsf{Prog} \mid P \lesssim_\beta \mathtt{rec}\ \mathtt{x}.\ M \wedge M[P/\mathtt{x}] \lesssim_\alpha P\}$$

It is an element because $M[\mathtt{rec}\ \mathtt{x}.\ M/\mathtt{x}] \approx \mathtt{rec}\ \mathtt{x}.\ M$. Suppose P is another element. Then

$$\{(N[\mathtt{rec}\,\mathtt{x}.\,M/\mathtt{x}], N[P/\mathtt{x}]) \mid \mathtt{x} \vdash N\}$$

is contained in $\gtrsim_\beta$ and, by Lemma 3, is a simulation up to $\lesssim_\alpha$ on the right. Lemma 1(1) tells us that it is contained in $\lesssim_\alpha$, so $\mathtt{rec}\ \mathtt{x}.\ M \lesssim_\alpha P$ as required. □

Corollary 1. *Let $\mathtt{x} \vdash M, M'$ be terms such that $M[P/\mathtt{x}] \approx M'[P/\mathtt{x}]$ for all programs P. Then* $\mathtt{rec}\ \mathtt{x}.\ M \approx \mathtt{rec}\ \mathtt{x}.\ M'$.

This result may also be proved using Howe's method [How96, Lev06].

4 Conclusions and Further Work

The present paper was greatly inspired by the denotational semantics in [Ros04], where recursion is interpreted by a "reflected" fixpoint calculated in two steps.

The quotient of CCS by bisimilarity is an image-countable transition system $\mathscr{S}$ in which bisimilarity is discrete. (It can also be described as a final coalgebra [TR98].) Therefore nesting fixpoints in $A(\mathscr{S})$ are genuine fixpoints and unique. This almost provides a denotational semantics, except that some monotone endofunctions do not have a nesting fixpoint. Perhaps restricting to the *exploratory* functions of [LW09] would be fruitful, as these are all definable in a sufficiently rich calculus.

In [Abr91] a domain theoretic model is provided that captures bisimilarity between processes without divergences. For general processes it induces a more subtle preorder.

The results of this paper may be adapted to lower (i.e. divergence-insensitive) applicative bisimulation [Abr90] in nondeterministic λ-calculus. However, in this instance Howe's method is stronger because it shows applicative bisimilarity to be preserved not only by recursion (Corollary 1) but also by application.

References

[Abr90] S. Abramsky. The lazy λ-calculus. In *Research topics in Functional Programming*, pages 65–117. Addison Wesley, 1990.

[Abr91] S Abramsky. A domain equation for bisimulation. *Information and Computation*, 92(2), 1991.

[GV92] Jan Friso Groote and Frits Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, October 1992.

[How96] D J Howe. Proving congruence of bisimulation in functional programming languages. *Inf. and Comp.*, 124(2), 1996.

[Las98] S B Lassen. *Relational Reasoning about Functions and Nondeterminism*. PhD thesis, Univ. of Aarhus, 1998.

[Lev06] P B Levy. Infinitary Howe's method. In *Proc., 8th Intl. Workshop on Coalgebraic Methods in Comp. Sci., Vienna*, volume 164(1) of *ENTCS*, 2006.

[LW09] Paul Blain Levy and Kidane Yemane Weldemariam. Exploratory functions on nondeterministic strategies, up to lower bisimilarity. *Electr. Notes Theor. Comput. Sci*, 249:357–375, 2009.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[Ros04] A W Roscoe. Seeing beyond divergence. presented at BCS FACS meeting "25 Years of CSP", July 2004.

[TR98] Daniele Turi and Jan J. M. M. Rutten. On the foundations of final coalgebra semantics. *Mathematical Structures in Computer Science*, 8(5):481–540, 1998.

[WN95] G. Winskel and M. Nielsen. Models for concurrency. In S. Abramsky, D. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*. Oxford University Press, 1995.