Bisimulation and coinduction: from the origins to today

Davide Sangiorgi

University of Bologna

Email: Davide.Sangiorgi@cs.unibo.it
http://www.cs.unibo.it/~sangio/

LICS'04

The talk

- bisimulation and coinduction: a short introduction
- origins (bisimulation, coinduction, fixed-point theorems, ...)

Thanks: J. van Benthem, E. Clarke, Y. Deng, F. Honsell, R. Milner, W.P. de Roever

Equality on processes, coinductively

Bisimulation:

A relation $\mathcal{R}$ s.t.

$$\begin{array}{ccc} P & \mathcal{R} & Q \\ \alpha\downarrow & & \downarrow\alpha \\ P' & \mathcal{R} & Q' \end{array}$$

Bisimilarity ($\sim$) :

$\bigcup \; \{\mathcal{R} \; : \; \mathcal{R}$ is a bisimulation $\}$

(coind. definition)

Hence:

$$\frac{x \;\mathcal{R}\; y \qquad \mathcal{R} \text{ is a bisimulation}}{x \sim y}$$

(coind. proof principle)

Coincidence

Locality + **no order** :

- **“simple” proofs**
- avoid the dangers of **circularity**

  cf: paradoxes, like Russel’s
- effective on **infinite objects**

  direct, natural modelling

Another examples of coinduction: divergence

A Rewriting System

D-invariant: a predicate $\mathcal{P}$ s.t.

$$\frac{M \in \mathcal{P} \qquad M \longrightarrow M'}{M' \in \mathcal{P}}$$

Divergence ($\Uparrow$) $\triangleq \bigcup \{\mathcal{P} \;:\; \mathcal{P}$ is a D-invariant $\}$

(coind. definition)

Hence:

$$\frac{M \in \mathcal{P} \qquad \mathcal{P} \text{ is a D-invariant}}{M \Uparrow}$$

(coind. proof principle)

Knaster-Tarski

On a **complete lattice** (i.e., a partial order $<$ with all joins)

Theorem On a complete lattice, a monotone endofunction F has a complete lattice of post-fixed points

x is **post-fixed point** of F if $x < F(x)$

Hence: **[coinduction proof principle, à la Knaster-Tarski]**

$$\frac{F \text{ monotone} \qquad x < F(x)}{x < \mathrm{gfp}\,(F)}$$

Knaster-Tarski

On a **complete lattice** (i.e., a partial order $<$ with all joins)

Theorem On a complete lattice, a monotone endofunction F has a complete lattice of post-fixed points

x is **post-fixed point** of F if $x < F(x)$

Hence: **[coinduction proof principle, à la Knaster-Tarski]**

$$\frac{F \text{ monotone} \qquad x < F(x)}{x < \mathrm{gfp}\,(F)}$$

To go deeper into coinduction: the **coalgebraic approach**

Knaster-Tarski sufficient for this talk

Origins

Two concepts here to be tracked down:

- **bisimulation**
- **coinductive principles**

  (only here the issues of circularity arise)

$$
\begin{array}{ccc}
P & \sim & Q \\
{\scriptstyle \alpha}\downarrow & & \downarrow{\scriptstyle \alpha} \\
P' & \sim & Q'
\end{array}
$$

$$
\frac{\Gamma, \langle p_1, q_1 \rangle \sim \langle p_2, q_2 \rangle \vdash p_i \sim q_i}{\Gamma \vdash \langle p_1, q_1 \rangle \sim \langle p_2, q_2 \rangle}
$$

3 lines, beginning early 70s

- **Computer Science**
- **Philosophical logic** (modal logic)
- **Set theory**

Common basis:

(weak) homomorphism between algebraic structures

Origins: bisimulation in CS

A celebrated paper:

D. Park, Concurrency and Automata on Infinite Sequences 5th GI (Gesellschaft fur Informatik) Conference, Karlsruhe, LNCS 104, Springer, 1981.

Classical and universal reference for bisimulation: correct?

David Park:

- **one of the pioneers of theoretical CS**
- **fundamental contributions in semantics** (fixed-point theory, fairness, ...)

Algebraic theory of automata, mid 60s

[cf: Ginzburg's book]

Homomorphism between (deterministic) Mealy automata

σ: transition function (partial)

A surjective function $\mathcal{F}$ s.t. $\forall$ input a:

(algebraically)	$\mathcal{F}\,\sigma_a \subseteq \sigma_a\,\mathcal{F}$	$[\sigma_a\,\mathcal{F}^{-1} \subseteq \mathcal{F}^{-1}\,\sigma_a]$
(operationally)	$\begin{array}{ccc} s & \overset{\mathcal{F}}{\longmapsto} & t \\ \sigma_a\downarrow & \searrow & \downarrow\sigma_a \\ s' & \underset{\mathcal{F}}{\longmapsto} & t' \end{array}$	

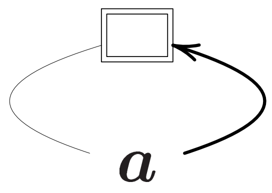
(similar conditions on the outputs produced)

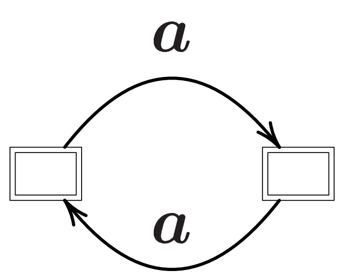
Theorem A minimal automata has no proper homomorphic image

$\mathrm{A} \leq \mathrm{B} \triangleq$ "B **can do at least the translations that** A **does**" (statewise)

Theorem homomorphism implies $\leq$

Converse: (very much) no





Weak homorphism

A total relation $\mathcal{R}$ s.t. $\forall$ input a:

(algebraically)	$\sigma_a \ \mathcal{R}^{-1} \subseteq \mathcal{R}^{-1} \ \sigma_a$
(operationally)	$\begin{array}{ccc} s & \mathcal{R} & t \\ \sigma_a \downarrow & \searrow & \downarrow \sigma_a \\ s' & \mathcal{R} & t' \end{array}$

Theorem weak homomorphism implies $\leq$

- more powerful theorem, but converse still false
- weak homomorphism stronger than simulation

What about

$A \sim B \triangleq$ **"A and B do exactly the same translations"**

?

Bisimulation might have been discovered

Milner, 1970

An algebraic definition of simulation between programs

2nd International Joint Conferences on Artificial Intelligence, London, 1971

- Programs: partial, sequential, imperative
- Program correctness
- When 2 programs realise the same algorithm?
- Milner’s proposal: weak homomorphism

(called simulation)

Novelties (towards bisimulation):

– **the operational meaning of weak homorphism**

– **the proof technique**

Used by Hoare:

Proof of Correctness of Data Representations, **Acta Informatica, 1, 271-281,** 1972

Park, 1981

Concurrency and Automata on Infinite Sequences LNCS 104, Springer, 1981

Topic: ω-regular languages

Main claimed contributions of the paper:

- characterisation as sets of recursive equations involving minimal and maximal fixed points
- variants of Buchi automata as recognisers
- closure under a fair concurrency operator

Bisimulation: presented as a proof technique for language equivalence

> "[Bisimulation is] **derived from Milner's notion of simulation**"

> "This provides motivation to be interested in proof principles for automata such as those involved here [cf: bisimulation] even though **their utility for the purposes of operational semantics of programs is obviously limited**."

No bisimilarity, no coinduction

Where do these come from then?

Milner, during the 70s

Developping a novel theory of processes (**CCS**) where **behavioural equivalence** is fundamental and based on **locality**

$$\begin{array}{ccc} P & \sim_{n+1} & Q \\ a\downarrow & & \downarrow a \\ P' & \sim_n & Q' \end{array}$$

$$\sim_0 \triangleq \text{Processess} \times \text{Processess}$$

$$\sim_\omega \triangleq \bigcap_n \sim_n$$

A Calculus of Communicating Systems LNCS 92, Springer, 1980

Lemma $\sim_\omega$ **is not invariant under transitions**

Park, 80/81: sabbatical in Edinburgh

- **Staying at Milner's (!)**
- **A fixed-point reading of Milner's theory:**

  The definition of $\sim_\omega$ is based on a functional $\mathcal{F}$ that is

  * monotone
  * non-cocontinuous

- **Applying fixed-point theory:**

  Bisimilarity $(\sim) \ \triangleq \ \mathrm{gfp}(\mathcal{F})$

  A bisimulation : a post-fixed point of $\mathcal{F}$

  Corollary : any bisimulation $\subseteq \sim$

  $\sim \ \triangleq \ \bigcap_{\lambda \text{ ordinal}} \mathcal{F}^{\lambda}(\mathrm{Processess} \times \mathrm{Processess})$

Milner:

- immediately accepts it
- makes it popular through his work on CCS

Reference:

D. Park slides for Bad Honnef Workshop on Semantics 1981

Moral:

> if you buy a big enough house you can benefit from other people's ideas
>
> — Milner

Why bisimulation has been discovered so late?

Example: automata theory

– weak homomorphism

– bisimulation implicit in standard algorithms from the 50s and the 60s

Algorithm for minimisation of automata

[Huffman 1954 and Moore 1956]

[also: the Myhill-Nerode theorem 1957-58]

Find the **non-equivalent states**, as an inductive set N:

1. If s **final** and t **is not**, then $s\ N\ t$
2. if $\exists\ a$ s.t. $\sigma(s,a)\ N\ \sigma(s,a)$ then $s\ N\ t$

The complement set: the **equivalent states**

What is this complement set?

The largest relation $\mathcal{R}$ s.t.

1. s **final** and $s \mathrel{\mathcal{R}} t$ imply t **final**, and the converse
2. $\forall\, a$, if $s \mathrel{\mathcal{R}} t$ then $\sigma(s, a) \mathrel{\mathcal{R}} \sigma(s, a)$

[cf: bisimilarity]

NB: any relation with 1-2 above relates equivalent states

[cf: bisimulation]

Origins: bisimulation in Philosophical logic

Modal logics interpreted on Kripke structures

Homomorphism (on models with a single relation $\longrightarrow$) **:**

a function $\mathcal{F}$ s.t.

$$\begin{tikzcd} s \arrow[r, mapsto, "\mathcal{F}"] \arrow[d] \arrow[dr, phantom, "\Searrow"] & t \arrow[d] \\ s' \arrow[r, mapsto, "\mathcal{F}"'] & t' \end{tikzcd}$$

Differences with bisimulation

– **no back condition**

– **functional**

When is the truth of a formula preserved when the model changes?

Theorem Modal formulas are not invariant under homomorphism

– **require an "iff" in the clause** (strong homomorphism)

– **require a back condition** (p-morphism)

[Jongt and Troelstra 1966, Segerberg 1971]

$$
\begin{tikzcd}
s \arrow[r, mapsto, "\mathcal{F}"] \arrow[d] \arrow[dr, phantom, "\Leftrightarrow"] & t \arrow[d] \\
s' \arrow[r, mapsto, "\mathcal{F}"'] & t'
\end{tikzcd}
$$

Theorem Modal formulas are invariant under p-morphism

A better invariance: **p-relations** (bisimulations)

[van Benthem, 76]

Theorem **(van Benthem)** A first-order logic formula (over Kripke structures) is equivalent to a modal formula iff it is bisimulation invariant

A better invariance: **p-relations** (bisimulations)

[van Benthem, 76]

Theorem **(van Benthem)** A first-order logic formula (over Kripke structures) is equivalent to a modal formula iff it is bisimulation invariant

The crux of the proof:

(Two states of) two models satisfy the same modal formulas iff their "saturated extensions" are bisimilar

cf: logical characterisation of bisimilarity **[Hennessy-Milner, 85]**

No coinduction, really

Origins: coinduction in Set Theory

Foundations of set theory (cf: non-well-founded sets)

– **Forti, Honsell** '80-83, **Hinnion** '80-81

  Bisimulations: f-conservative relations, contractions

  Coinduction?

  * yes
  * a little hidden (more attention to bisimulation equivalences than bisimulations)

– **Aczel** '85-89

  nwf sets popular, motivated by Milner's work on CCS

  the basis of the coalgebraic approach to semantics

The origins: coinduction and fixed-points in CS

Greatest fixed-points

– **David Park**

fairness of the merge of two infinite sequences

around mid 70s: notes, a number of talks

First application of gfp in CS??

> On the Semantics of Fair Parallelism, Copenhagen Winter School, LNCS 86, Springer, 1979

(first publication?)

– **Edmund M. Clarke**

Program invariants as fixed points Preliminary report: 18th Foundations of Computer Science, IEEE, 1977

Final paper: Computing, 21 : 273-294 1979

Also in: Clarke's PhD thesis 1976

Program invariants as gfp's

Soundness and completeness for Floyd-Hoare systems

Soundness proved using coindution

First use of coinduction principle in CS??

– **Willem P. de Roever**

On backtracking and greatest fixpoints Formal description of Programming Concepts, North-Holland, 1978

First time coinduction principle proposed as a proof technique??

(called gfp induction)

Coinduction implicit in a number of earlier works. Examples:

- Unification

  Structural equivalence of graphs **[Hopcroft and Ullman '71]**

- Invariant properties (60s, 70s)

  A huge literature

  Floyd-Hoare Axiom systems, weakest preconditions

Least fixed-points in CS

D. Scott, J.W. de Bakker, A Theory of Programs, Handwritten notes, Wien, 1969

H. Bekic Definable operations in general algebras, and the theory of automata and flowcharts, IBM Lab., Wien, 1969

D. Park Fixpoint induction and proofs of program properties Machine Intelligence, Edinburgh University Press 1969

They show the importance of lfp's and Knaster-Tarski in programming languages

Various rules for reasoning about lfp's

Important earlier work:

Landin and McCarthy [early 60s], Floyd [1967]

Earlier uses of least fixed-points:

– recursion theory **[Kleene 1958, Rogers, 1967]**

– formal languages **[Ginsburg and Rice 1962]**

The origins: fixed-point theorems

Knaster-Tarski (Knaster? Tarski?)

Bronislaw Knaster, 1928

Theorem **Every monotone function F over sets s.t. there is a set A with $F(A) \subseteq A$ has a fixed point D with $D \subseteq A$**

> Un theoreme sur les fonctions d'ensembles, Ann. Soc. Polon. Math. 6, 133-134, 1928

(the theorem is joint work with Tarski)

Alfred Tarski, 1939

Theorem **On a complete lattice, a monotone endofunction F has a complete lattice of fixed points**

discussed in some public lectures

> A lattice-theoretical fixpoint theorem and its applications (preliminary version), Bulletin AMS, vol. 55, 1051–1052, 1949

> A lattice-theoretical fixpoint theorem and its applications, Pacific Journal of Mathematics, vol. 5, 285–309, 1955

Closely related work:

L. Kantorovich, The method of successive approximations for functional equations, Acta Mathematica v.71, 63-97, 1939

Should the **Knaster-Tarski** theorem be called the **Knaster-Tarski-Kantorovich** theorem?

NB: Leonid Vitalevich Kantorovich (1912-1986), Russian mathematician, 1975 Nobel Prize winner in Economics

Other fixed-point theorems

Theorem **For a continuous endofunction F in a complete lattice: lfp(F) = $\bigcup_n F^n(\perp)$**

Late 30s

Kleene? Tarski? Kantorovitch?

Sometimes continuity too strong (example: non-determinism)

Theorem **For a monotone endofunction F in a complete lattice: lfp(F) = $\bigcup_{\lambda \text{ ordinal}} F^{\lambda}(\perp)$**

Hitchcock and Park 1972

Similar results in mathematics existed earlier

Some curiosities on names

- **Bisimulation** $\Rightarrow$ **Milner 1980 ('81?)**

  Park's proposal: **mimicry**

- **Coinduction** $\Rightarrow$ **Barwise 1985** (?)

  made popular by **Milner and Tofte 1991**