Revisiting: Algebraic laws for nondeterminism and concurrency

Matthew Hennessy

Milner-Symposium, Edinburgh  April 2012
History of a paper

Algebraic laws for nondeterminism and concurrency, JACM 1985
Matthew Hennessy and Robin Milner

- Research in late 1979 33 years ago
- Results presented at ICALP 1980 32 years ago
  (On Observing Nondeterminism and Concurrency)
- Rejected for publication 1982
- Rejected for publication 1983
- Published in JACM 1985
Edinburgh 1979  33 years ago

- No Labelled Transition Systems
- No CCS  No CSP  No ACP  No ...
- No street lightening
- What happened to the sun?
- Lots of mushrooms
- No Bisimulations
- When does the summer arrive?
- Walks on Arthurs seat
- Lots of parking near George Square
- .......
- .......
- .......
- .......
- .......

Edinburgh 1979: Lots of denotational semantics

\[ D \cong [D \to D] \quad \text{functions} \quad \text{Scott, 1969} \]

\[ P \cong V \to (V \times P) \quad \text{transformers} \quad \text{Milner 1973} \]

\[ R \cong \mathcal{P}(S_\bot + (\mathcal{P}(S_\bot) \otimes R_\bot))^S \quad \text{resumptions} \quad \text{Plotkin 1976} \]

\[ P_L \cong \mathcal{P}(\sum_{\beta \in L}(U_\beta \times (V_\beta \to P_L))) \quad \text{processes} \quad \text{Milne & Milner 1979} \]
Edinburgh 1979: Lots of algebraic semantics

The Auld Alliance

- Jean-Marie Cadiou (1972): Recursive Definitions of Partial Functions and their Computations
- Jean Vuillemin (1973): Proof Techniques for Recursive Programs
- Bruno Courcelle, Maurice Nivat (1978): The Algebraic Semantics of Recursive Programme Schemes
- Irene Guessarian (1981): Algebraic Semantics
Edinburgh 1979: Lots of algebraic semantics

The Auld Alliance

- Jean-Marie Cadiou (1972): Recursive Definitions of Partial Functions and their Computations
- Jean Vuillemin (1973): Proof Techniques for Recursive Programs
- Bruno Courcelle, Maurice Nivat (1978): The Algebraic Semantics of Recursive Programme Schemes
- Irene Guessarian (1981): Algebraic Semantics

- Magmas: ordered sets with operators
- Ideal completions: adding limit points
- Initial algebra semantics
A behavioural equivalence

ICALP 1980:

over $P$. Since in general there may be various means of communication we have a set of relations $\{R_i \in P \times P, i \in I\}$. Using these atomic experiments, we define a sequence of equivalence relations $\sim_n$ over $P$ as follows:

Let $p \sim_0 q$ if $p,q \in P$

$p \sim_{n+1} q$ if

1) $\forall i \in I, \forall p', p' \in R_i$ implies $\exists q'. \langle q, q' \rangle \in R_i \land p \sim_n q'$

and 2) $\forall i \in I, \langle q, q' \rangle \in R_i$ implies $\exists p'. \langle p, p' \rangle \in R_i \land p \sim_n q'$

Then $p$ is observationally equivalent to $q$, written $p \sim q$, if $p \sim_n q$ for every $n$. 
Observational equivalence

- Reduction semantics: $P \rightarrow Q$
  well-known

- Observational semantics: $P \mu \rightarrow Q$
  new to me

Observing processes:

- $p \sim o q$ for all $p, q$
  zero observations
- $p \sim n + 1 q$ if for every $\mu (n + 1)$ observations

(i) $p \mu \rightarrow p'$ implies $q \mu \rightarrow q'$ such that $p' \sim n q'$
(ii) $q \mu \rightarrow q'$ implies $p \mu \rightarrow p'$ such that $p' \sim n q'$

Transfer properties

Observational equivalence:

$p \sim q$ if $p \cap n \geq 0 \sim n \sim q$
Observational equivalence

- Reduction semantics: \( P \rightarrow Q \) well-known
- Observational semantics: \( P \xrightarrow{\mu} Q \) new to me
Observational equivalence

- Reduction semantics: \( P \rightarrow Q \) well-known
- Observational semantics: \( P \xrightarrow{\mu} Q \) new to me

Observing processes:

- \( p \sim_o q \) for all \( p, q \) zero observations
- \( p \sim_{n+1} q \) if for every \( \mu \) (\( n + 1 \) observations)
  (i) \( p \xrightarrow{\mu} p' \) implies \( q \xrightarrow{\mu} q' \) such that \( p' \sim_n q' \)
  (ii) \( q \xrightarrow{\mu} q' \) implies \( p \xrightarrow{\mu} p' \) such that \( p' \sim_n q' \)

Transfer properties
Observational equivalence

Observational equivalence:

- Reduction semantics: $P \rightarrow Q$ (well-known)
- Observational semantics: $P \mu \rightarrow Q$ (new to me)

Observing processes:

- $p \sim_o q$ for all $p$, $q$ (zero observations)
- $p \sim_{n+1} q$ if for every $\mu$ ( $(n + 1)$ observations)
  
  (i) $p \mu \rightarrow p'$ implies $q \mu \rightarrow q'$ such that $p' \sim_n q'$
  
  (ii) $q \mu \rightarrow q'$ implies $p \mu \rightarrow p'$ such that $p' \sim_n q'$

Transfer properties

Observational equivalence:

$$p \sim q \text{ if } p \left( \bigcap_{n \geq 0} \sim_n \right) q$$
Observing processes

Life could get much more complicated:
Observing processes

\[
P_2 \sim_o Q_2 \quad P_2 \sim_1 Q_2 \quad P_2 \sim_2 Q_2 \quad P_2 \not\sim_3 Q_2
\]
Observing processes

$P_2 \sim_o Q_2$

$P_2 \sim_1 Q_2$

$P_2 \sim_2 Q_2$

$P_2 \sim/3 Q_2$

Life could get much more complicated:

$P_n \sim_n Q_n$

$P_n \sim/(n+1) Q_n$
Observational equivalence: Where from?

A Denotational Model

\[ P_L \cong P\left( \sum_{\beta \in L} (U_\beta \times (V_\beta \rightarrow P_L)) \right) \]

- \( L \): set of ports
- \( U_\beta \): output values on port \( \beta \)
- \( V_\beta \): input values on port \( \beta \)

A simplification \( U_\beta = V_\beta = 1 \):

\[ P_L \cong P\left( \sum_{\mu \in L} P_L \right) \]

How would you compare two elements \( p, q \) from \( P_L \)?
Observational equivalence: a theorem

ICALP 1980:

Then $p$ is observationally equivalent to $q$, written $p \sim q$, if $p \sim^n q$ for every $n$. Before discussing $\sim$ we give some of its properties. For any $S \subseteq P \times P$ let $E(S)$ be defined by

$$(p, q) \in E(S) \text{ if } \forall i \in I$$

i) $(p, p') \in R_i \Rightarrow \exists q'. (q, q') \in R_i, (p', q') \in S$  

ii) $(q, q') \in R_i \Rightarrow \exists p'. (q, q') \in R_i, (p', q') \in S$

We say that a relation $R$ is image-finite if for each $p$, $\{p' | (p, p') \in R\}$ is finite.

Theorem 2.1

If each $R_i$ is image-finite then $\sim$ is the maximal solution to $S = E(S)$. \[\square\]
First research experiment

Process language:

\[ p \in W_{\Sigma_1} ::= 0 \mid p + p \mid \mu.p \]

finite non-deterministic machines
First research experiment

**Process language:**

\[ p \in W_{\Sigma_1} ::= 0 \mid p + p \mid \mu.p \]

Result:

- \[ \bigcap_{n \geq 0} (\sim_n) \] is a \(\Sigma_1\)-congruence

- \[ p \bigcap_{n \geq 0} (\sim_n) q \iff p =_A q \]

**Axioms (A):**

- \[ x + (y + z) = (x + y) + z \]
- \[ x + x = x \]
- \[ x + y = y + x \]
- \[ x + 0 = x \]

finite non-deterministic machines
First research experiment

Process language:

\[ p \in W_{\Sigma_1} ::= 0 \mid p + p \mid \mu.p \]

Result:

- \( \bigcap_{n \geq 0} (\sim_n) \) is a \( \Sigma_1 \)-congruence

- \( p \bigcap_{n \geq 0} (\sim_n) q \) iff \( p =_A q \)

Axioms (A):

\[
\begin{align*}
& x + (y + z) = (x + y) + z \\
& x + x = x \\
& x + y = y + x \\
& x + 0 = x
\end{align*}
\]

Denotational semantics:

\[ p \bigcap_{n \geq 0} (\sim_n) q \] iff \( \llbracket p \rrbracket_{(W_{\Sigma_1} \setminus A)} = \llbracket q \rrbracket_{(W_{\Sigma_1} \setminus A)} \)

\((W_{\Sigma_1} \setminus A) : \text{Initial algebra over } W_{\Sigma_1} \text{ generated by axioms } A \)
Robin had a lot of background

- 1978: Algebras for Communicating Systems
- 1978: Synthesis of Communicating Behaviour
- 1978: Flowgraphs and Flow Algebras
- 1979: An Algebraic Theory for Synchronisation
- 1979: Concurrent Processes and Their Syntax
Robin had a lot of background

- 1978: Algebras for Communicating Systems
- 1978: Synthesis of Communicating Behaviour
- 1978: Flowgraphs and Flow Algebras
- 1979: An Algebraic Theory for Synchronisation
- 1979: Concurrent Processes and Their Syntax

Combinators and their Laws proposed:
Robin had a lot of background

- 1978: Algebras for Communicating Systems
- 1978: Synthesis of Communicating Behaviour
- 1978: Flowgraphs and Flow Algebras
- 1979: An Algebraic Theory for Synchronisation
- 1979: Concurrent Processes and Their Syntax

Combinators and their Laws proposed:

- Flowgraphs and flow algebras for static structure
- Synchronisation trees for dynamics
Justifying equations

Flowgraphs:

Let \( p = \sum_i \lambda_i p_i \), \( q = \sum_j \mu_j q_j \). Then

\[
\begin{align*}
(p | q) &= \sum_i \lambda_i (p_i | q) + \sum_j \mu_j (p | q_j) + \sum \mu_j \lambda_i \tau.
\end{align*}
\]
Justifying equations

Flowgraphs:

Synchronisation trees:
Let \( p = \sum_i \lambda_i \cdot p_i \), \( q = \sum_j \mu_j \cdot q_j \). Then

\[
p|q = \sum_i \lambda_i \cdot (p_i|q) + \sum_j \mu_j \cdot (p|q_j) + \sum_{\mu_j = \lambda_i} \tau \cdot (p_i|q_j)
\]
Theorems for free

$\Sigma_2 = \Sigma_1$ plus

- Parallelism: $|$
- Restriction: $\setminus \lambda$
- Renaming: $[S]$ $S$ a function over names

Result:

- $\left( \cap_{n \geq 0} \sim_n \right)$ is a $\Sigma_2$ congruence

- $p \left( \cap_{n \geq 0} \sim_n \right) q$ iff $p =_{A2} q$
Theorems for free

\[ \Sigma_2 = \Sigma_1 \text{ plus} \]

- Parallelism: \(|\)
- Restriction: \(\backslash \lambda\)
- Renaming: \([S]\) \(S\) a function over names

Result:

- \(\left( \cap_{n \geq 0} \sim_n \right)\) is a \(\Sigma_2\)- congruence

- \(p \left( \cap_{n \geq 0} \sim_n \right) q\) \iff \(p = A_2 q\)

\[ A_2 = A_1 + \text{existing axioms for } |, \backslash \lambda, [S] \]
Weak case: abstracting from internal activity $\tau$

- Weak observational semantics:

  $$ P \xrightarrow{\mu} Q \text{ meaning } P \xrightarrow{\tau} \ast \xrightarrow{\mu} \ast \xrightarrow{\tau} \ast Q $$

External observations:

- $p \approx_{o} q$ for all $p, q$  
  zero observations

- $p \approx_{n+1} q$ if for every $\mu \in \text{Act}_\tau$
  (n + 1) observations

  (i) $p \xrightarrow{\mu} p'$ implies $q \xrightarrow{\mu} q'$ such that $p' \approx_n q'$
  (ii) $q \xrightarrow{\mu} q'$ implies $p \xrightarrow{\mu} p'$ such that $p' \approx_n q'$

Weak transfer properties

look: no hats
Weak case: abstracting from internal activity $\tau$

- Weak observational semantics:

  \[ P \xrightarrow{\mu} Q \text{ meaning } P \xrightarrow{\tau} * \xrightarrow{\mu} \tau \xrightarrow{\star} * Q \]

External observations:

- $p \approx_o q$ for all $p, q$
- $p \approx_{n+1} q$ if for every $\mu \in \text{Act}_\tau$

  (i) $p \xrightarrow{\mu} p'$ implies $q \xrightarrow{\mu} q'$ such that $p' \approx_n q'$
  (ii) $q \xrightarrow{\mu} q'$ implies $p \xrightarrow{\mu} p'$ such that $p' \approx_n q'$

Weak transfer properties

look: no hats

Weak observational equivalence:

\[ p \approx q \text{ if } p \left( \bigcap_{n \geq 0} \approx_n \right) q \]
Equational characterisation

Problem: \( \bigcap_{n \geq 0} \approx_n \) is NOT preserved by operators + or |
Equational characterisation

- Problem: \((\cap_{n \geq 0} \approx_n)\) is NOT preserved by operators + or | 
- Result: In \(\Sigma_1\), \(p(\cap_{n \geq 0} \approx_n)c\) q iff \(p =_{WA1} q\)

Axioms WA1: add to A1 the \(\tau\)-axioms:
\[x + \tau.x = \tau.x\]
\[\mu.((x + \tau.y) = \mu.(x + y) + \mu.y \quad \mu.\tau.y = \mu.y\]
\[\mu.(x + \tau.y) = \mu.(x + \tau.y) + \mu.y\]
Equational characterisation

- Problem: \(\bigcap_{n\geq 0} \approx n\) is NOT preserved by operators + or |.
- Result: In \(\Sigma_1\), \(p \bigcap_{n\geq 0} \approx n\) \(c\) iff \(p =_{WA1} q\).

Axioms WA1: add to A1 the \(\tau\)-axioms:

\[
\begin{align*}
    x + \tau \cdot x &= \tau \cdot x \\
    \mu \cdot (x + \tau \cdot y) &= \mu \cdot (x + y) + \mu \cdot y \\
    \mu \cdot \tau \cdot y &= \mu \cdot y \\
    \mu \cdot (x + \tau \cdot y) &= \mu \cdot (x + \tau \cdot y) + \mu \cdot y
\end{align*}
\]

Where did these come from?
An exercise in Behaviour Algebra notes by Robin on modelling queues

\[ \mu.\tau.\mathbb{X} = \mu.\mathbb{X} \] (\tau 1)

get the required result (17) from (18), we shall need our final extra behaviour law.

which case that a \(\tau\) guard may be absorbed in a guarded...
An exercise in Behaviour Algebra notes by Robin on modelling queues

get the required result (17) from (18), we shall need our first extra behaviour law

\[ \text{For any guard } \mu, \mu \cdot \tau \cdot X = \mu \cdot X \]  

(\tau 1)

which case that a \( \tau \) guard may be absorbed in a guarded

\[(B) J \vdash \phi. \text{ Here we shall need two extra behaviour laws}

\[ X + \tau \cdot X = \tau \cdot X \]  

(\tau 2)

\[ X + X = X \]  \hspace{1cm} \text{(idempotence)}.

They have together the important corollary

\[ X + \tau \cdot (X + Y) = \tau \cdot (X + Y) \]  

(\tau 2')

\[(B1) \text{ If } b \neq 0, \text{ we get from (21)}

q(s) \equiv \text{queue}_{i+1}(s) + \sum_{j \in J} \tau \cdot \text{queue}_{i+1}(S_j)\]
Hennessy Milner Logic

where did this come from?

Observational equivalence $p \ (\bigcap_{n \geq 0} \sim_n) \ q$

- *Inspired by identity in domain* $P_L \cong \mathcal{P}(\sum_{\mu \in L} P_L)$
Hennessy Milner Logic  where did this come from?

Observational equivalence $p \quad (\bigcap_{n \geq 0} \sim_n) \quad q$

- *Inspired by* identity in domain $P_L \cong \mathcal{P}(\sum_{\mu \in L} P_L)$
- *Requires* independent justification

Why are these behaviourally different:

![Behavioral differences](image)
Hennessy Milner Logic where did this come from?

Observational equivalence $p \ (\cap_{n \geq 0} \sim_n) \ q$

▶ *Inspired by* identity in domain $P_L \cong \mathcal{P}(\sum_{\mu \in L} P_L)$
▶ *Requires* independent justification

Why are these behaviourally different:

Discover difference using *interaction games*:

▶ can do action $x$
▶ can not do action $x$
Discovering differences

Q_2 can perform a so that every time a is subsequently performed both b and c can be performed
Discovering differences

\[ P_2 \]

\[ Q_2 \]

\[ Q_2 \text{ can perform } a \text{ so that every time } a \text{ is subsequently performed both } b \text{ and } c \text{ can be performed} \]

\[ Q_2 \models \langle a \rangle[a](\langle b \rangle tt \land \langle c \rangle tt) \]

\[ P_2 \not\models \ldots \]
Hennessy Milner Logic

\[ A, B \in \mathcal{L} ::= \text{tt} \mid A \land B \mid \neg A \mid \langle \mu \rangle A \]

- \( p \models \langle \mu \rangle A \) if \( p \xrightarrow{\mu} p' \) such that \( p' \vdash A \)
- \( p \models A \land B \) if \ldots .

Result:

- \( p \models (\bigcap_{n \geq 0} \sim_n) q \iff \mathcal{L}(p) = \mathcal{L}(q) \)
- \( p \models (\bigcap_{n \geq 0} \sim_n) q \iff p \models A \) and \( q \not\models A \), for some \( A \in \mathcal{L} \).

\( A \) is an explanation of why \( p, q \) are different.
Enter . . . David Park 1935 - 1990

Fixpoint induction:

If \( F(H) \leq H \) then \( \min X. F(X) \leq H \) requires monotonicity

Fair merge:

1979

\[ \text{fairmerge} = \max X. \min Y. (Fm(\min Z. Fm(Z), X), Y) \]

where \( Fm(X, Y) = \{(\epsilon, x, x) | x \in \Sigma \} \cup \{(x, \epsilon, x) | x \in \Sigma \} = \{(ax, y, az) | a \in \Sigma, (x, y, z) \in X\} = \{(x, ay, az) | a \in \Sigma, (x, y, z) \in Y\} \]
Fixpoint induction:

If $F(H) \leq H$ then $\min X. F(X) \leq H$
Fixpoint induction:

If $F(H) \leq H$ then $\min_X F(X) \leq H$

1970 machine intelligence

requires monotonicity

Fair merge:

$\text{fairmerge} = \max_X \min_Y (Fm(\min_Z Fm(Z, X), Y))$

1979

where $Fm(X, Y) = \{(\epsilon, x, x)|x \in \Sigma^\infty\} \cup \{(x, \epsilon, x)|x \in \Sigma^\infty\}$

$= \{(ax, y, az)|a \in \Sigma, (x, y, z) \in X\}$

$= \{(x, ay, az)|a \in \Sigma, (x, y, z) \in Y\}$
Using Maximal Fixpoints

Icalp 1980: Hennessy & Milner

Extensive use in meta-theory of processes:

- **Theorem 2.1** If each $R_i$ is image-finite then $\sim$ is the maximal solution to $S = E(S)$
- ALNC, page 157: Now let $\approx'$ be the maximal solution to the equation $S = E'(S)$
Using Maximal Fixpoints

Icalp 1980: Hennessy & Milner

Extensive use in meta-theory of processes:

- **Theorem 2.1** If each $R_i$ is image-finite then $\sim$ is the maximal solution to $S = E(S)$

- **ALNC, page 157:** Now let $\approx'$ be the maximal solution to the equation $S = E'(S)$

David Park:

Use maximal fixpoints in object-theory of processes

Replace $\bigcap_{n \geq 0} \sim_n$ with a maximal fixpoint $\sim_{bis}$
Co-induction à la David Park

Transfer property:
For $R \subseteq P \times P$, define $B(R) \subseteq P \times P$ by $p B(R) q$ whenever

(i) $p \xrightarrow{\mu} p'$ implies $q \xrightarrow{\mu} q'$ such that $p R q$
(ii) $q \xrightarrow{\mu} q'$ implies $p \xrightarrow{\mu} p'$ such that $p R q$

Bisimulations:

$R \subseteq P \times P$ is a bisimulation if $B(R) \subseteq R$

$p \sim_{bis} q$ if $p R q$ for some bisimulation $R$

Elegant proof for establishing $p \sim_{bis} q$
Co-induction à la David Park

Robin Milner: *A Calculus of Communicating Systems*, LNCS 1980
Co-induction à la David Park

Robin Milner: *A Calculus of Communicating Systems*, LNCS 1980


- elegant theory
- lots of worked examples
- detailed proofs
Proposed Theorem: In Lambda, if $FA \sqsubseteq A$ then $YF \sqsubseteq A$

Question: What is $\sqsubseteq$?
Jim Morris and his style of equivalences


- Proposed Theorem:
  In Lambda, if $FA \sqsubseteq A$ then $\mathbf{YF} \sqsubseteq A$
Proposed Theorem:
In Lambda, if $FA \sqsubseteq A$ then $YF \sqsubseteq A$

Question: What is $\sqsubseteq$?
Proposed Theorem:
In Lambda, if $FA \sqsubseteq A$ then $\Upsilon F \sqsubseteq A$

Question: What is $\sqsubseteq$ ?

Morris Preorder:

$A \sqsubseteq_{\text{morris}} B$ if for every context $C[\ ]$

$C[A]$ has a normal form implies $C[B]$ has a normal form
Morris - style of equivalences

Ingredients:

- A reduction semantics: $P \rightarrow Q$
- Results: $P \Downarrow v$
- Language syntax for contexts $C[]$

Contextual equivalence:

$P \simeq_{\text{cxt}} Q$ if for every context, for every barb,

$$C[P] \rightarrow^* P' \Downarrow v \iff C[Q] \rightarrow^* Q' \Downarrow v$$
Morris - style of equivalences

Ingredients:
- A reduction semantics: \( P \rightarrow Q \)
- Results: \( P \downarrow \nu \)
- Language syntax for contexts \( C[\] \)

Contextual equivalence:

\( P \simeq_{\text{cxt}} Q \) if for every context, for every barb,

\[
C[P] \rightarrow^* P' \downarrow \nu \quad \text{iff} \quad C[Q] \rightarrow^* Q' \downarrow \nu
\]

Where are the quantifiers?
Justifying Bisimulation Equivalence

Barbed congruence:
For image-finite CCS processes,

\[ P \cong_{bism} Q \iff P \cong_{barb} Q \]

Milner, Sangiorgi 1992
Justifying Bisimulation Equivalence

Barbed congruence:
For image-finite CCS processes,

\[ P \approx_{bism} Q \iff P \approx_{barb} Q \]

Reduction barbed congruence:
For arbitrary CCS processes,

\[ P \approx_{bism} Q \iff P \approx_{rbc} Q \]
Justifying Bisimulation Equivalence

Barbed congruence:
For image-finite CCS processes,

\[ P \approx_{bism} Q \iff P \approx_{barb} Q \]

Reduction barbed congruence:
For arbitrary CCS processes,

\[ P \approx_{bism} Q \iff P \approx_{rbc} Q \]

Both contextual equivalences are reduction closed:

1. \( P \rightarrow^* P' \) implies \( Q \rightarrow^* Q' \) s.t. \( P' \approx Q' \)
2. \( Q \rightarrow^* Q' \) implies \( P \rightarrow^* Q' \) s.t. \( P' \approx Q' \)
Bisimulations in the Modern World

Pick your favourite process language
Bisimulations in the Modern World

Pick your favourite process language

- Bisimulations do not provide a behavioural theory of processes *per se*
- Bisimulations provide a proof methodology for demonstrating processes to be equivalent
- HML provide a methodology for explaining why processes are not equivalent
Bisimulations in the Modern World

Pick your favourite process language

- Bisimulations do not provide a behavioural theory of processes *per se*
- Bisimulations provide a proof methodology for demonstrating processes to be equivalent
- HML provide a methodology for explaining why processes are not equivalent
- Bisimulations are very often sound w.r.t. the natural contextual equivalence $\approx_{\text{cxt}}$
- Bisimulations are sometimes complete w.r.t. the natural contextual equivalence $\cong_{\text{cxt}}$
- Formulating complete bisimulations very often sheds light process behaviour
Examples a very small sample

- **Asynchronous Picalculus:** Honda, Tokoro 1991, Amadio Castellani Sangiorgi 1998
- **Mobile Ambients:** Merro, Zappa Nardelli 2005
- **Existential and recursive types in lambda-calculus:** Sumii, Pierce 2007
- **Higher-order processes:** environmental bisimulations Sangiorgi, Kobayashi, Sumii 2007
- **Aspects in a functional language:** open bisimulations Jagadeesan, Pitcher, Riely 2007
- **Concurrent Probabilistic processes:** Deng, Hennessy 2011
Examples  a very small sample

- Mobile Ambients: Merro, Zappa Nardelli 2005
- Existential and recursive types in lambda-calculus: Sumii, Pierce 2007
- Concurrent Probabilistic processes: Deng, Hennessy 2011
- Bigraphs: Robin and co-workers
  - Bigraphs: all encompassing descriptive language
  - Recovery of LTS from reduction semantics
  - ensuring soundness of bisimulations