Robin Milner, 1934-2010
Concurrency: interaction, bisimulation, naming

Alan Jeffrey
Bell Labs, Enabling Computing Technologies Research
January 2011, ACM Principles of Programming Languages
Interaction

Lecture Notes in Computer Science

Edited by G. Goos and J. Hartmanis

Robin Milner

A Calculus of Communicating Systems

Springer-Verlag
Berlin Heidelberg New York
Concurrency = automata + interaction

I had an idea that automata theory was inadequate, because it hadn’t said what it was for two automata to interact with each other.

All quotes from Martin Berger’s An Interview With Robin Milner, 2003.
A Calculus of Communicating Systems

- Existing and simultaneous work on communication and processes.
  - Petri Nets (1962).
  - Lamport (1978) partially ordered events.
  - Pnueli (1979) temporal logic.
  - Hennessy and Plotkin (1979) resumptions.
  - ...
A Calculus of Communicating Systems

- Existing and simultaneous work on communication and processes.
- Labelled transition systems.
  - Automata where every state is an acceptor.
  - Alphabet of actions $\ell$, coactions $\bar{\ell}$ and $\tau$.
  - Support interaction:

$$
\begin{align*}
  P \xrightarrow{\ell} P' & \quad Q \xrightarrow{\bar{\ell}} Q' \\
  P | Q \xrightarrow{\tau} P' | Q'
\end{align*}
$$
A Calculus of Communicating Systems

- Existing and simultaneous work on communication and processes.
- Labelled transition systems.
- Process language.
  - Small, inductively defined syntax with recursive processes.
  - Inspired by language of regular expressions; includes interaction.
  - Semantics given operationally (cf. Plotkin) as an LTS.
A Calculus of Communicating Systems

- Existing and simultaneous work on communication and processes.
- Labelled transition systems.
- Process language.
- Inductively defined equivalence of processes.
  - $P \sim_0 Q$.
  - $P \sim_{n+1} Q$ whenever, for all $\alpha$:
    - If $P \xrightarrow{\alpha} P'$ then, for some $Q'$, $Q \xrightarrow{\alpha} Q'$ and $P' \sim_n Q'$.
    - If $Q \xrightarrow{\alpha} Q'$ then, for some $P'$, $P \xrightarrow{\alpha} P'$ and $P' \sim_n Q'$.
  - $P \sim_\omega Q$ whenever, for all $n$, $P \sim_n Q$.

Now called stratified bisimulation.
A Calculus of Communicating Systems

- Existing and simultaneous work on communication and processes.
- Labelled transition systems.
- Process language.
- Inductively defined equivalence of processes.
  - \( P \sim_0 Q \).
  - \( P \sim_{n+1} Q \) whenever, for all \( \alpha \):
    - If \( P \xrightarrow{\alpha} P' \) then, for some \( Q' \), \( Q \xrightarrow{\alpha} Q' \) and \( P' \sim_n Q' \).
    - If \( Q \xrightarrow{\alpha} Q' \) then, for some \( P' \), \( P \xrightarrow{\alpha} P' \) and \( P' \sim_n Q' \).
  - \( P \sim_\omega Q \) whenever, for all \( n \), \( P \sim_n Q \).

Now called stratified bisimulation. Implicit in earlier work:
- Moore's algorithm for DFA minimization (1956).

But now a first-class citizen, not a proof or algorithm technicality.
A Calculus of Communicating Systems

- Existing and simultaneous work on communication and processes.
- Labelled transition systems.
- Process language.
- Inductively defined equivalence of processes.
- Results.
  - Stratified bisimulation is a congruence.
  - Sound and complete axiomatization.
  - Logical characterization (Hennessy-Milner logic).
A Calculus of Communicating Systems

- Existing and simultaneous work on communication and processes.
- Labelled transition systems.
- Process language.
- Inductively defined equivalence of processes.
- Results.
- Spawned a new research area.
  - Papers, books, conferences, research networks.
  - Helped establish operational reasoning.
Bisimulation

4.1 Experimenting upon agents
To explain our approach, consider the case in which two agents A and B interact, whose defining equations are as follows:

- $A \Downarrow a_A$
- $B \Downarrow a_B$
- $A \Downarrow c \Rightarrow a_A \Downarrow c \Rightarrow a_B$
- $B \Downarrow c \Rightarrow a_B \Downarrow c \Rightarrow a_A$

Then, $A$ and $B$ may be thought of as finite-state automata over the alphabet $\mathcal{A}$, and their transition graphs are as follows:

- $A \Downarrow a_A \rightarrow A_1$
- $B \Downarrow a_B \rightarrow B_1$
- $A_1 \Downarrow c \Rightarrow a_A \Downarrow c \Rightarrow a_B \rightarrow A_2$
- $B_1 \Downarrow c \Rightarrow a_B \Downarrow c \Rightarrow a_A \rightarrow B_2$

Now, in standard automaton theory, an automaton is interpreted as a language, $L$, in a set of states over the alphabet $\mathcal{A}$. In the terminology of automata and languages, $A_1$ and $B_1$ are states for the language $L_1$, which we may write as $L_1 = \{ a_1, a_2 \}$. Similarly, for the language $L_2$, we may write $L_2 = \{ b_1, b_2 \}$.

We define $A$ and $B$ to be bisimilar (in the terminology of automata) if they are bisimilar, i.e., $A \Downarrow a_A \Rightarrow B \Downarrow a_B \Rightarrow A \Downarrow a_A \Rightarrow B \Downarrow a_B$.

**Definition:** $A$ and $B$ are bisimilar if for every state $a_A \in A$, there is a state $b_B \in B$ such that $A \Downarrow a_A \Rightarrow B \Downarrow b_B \Rightarrow A \Downarrow a_A \Rightarrow B \Downarrow b_B$.

**Example 1:** In Section 3.3 we discussed two agents $C_1$ and $D_1$, which are represented by the following transition graphs:

```
C_1
```

```
D_1
```

One can easily check that $C_1 = (C_2, B, (C_1, D_1), (C_2, D_1), (C_1, D_2))$ is a bisimulation.
The most important breakfast in computer science?

David Park was living in my house, reading my book, and he came down at breakfast time and said "There is something wrong with this... this isn't a maximal fixed point."
Technical problem: stratified bisimulation is not a fixed point.

- $\sim_\omega$ is not always $\sim_{\omega+1}$.
- Technical fix: transfinite induction.
Technical problem: stratified bisimulation is not a fixed point.

More importantly, missed an important proof technique.
- A relation $R$ is a bisimulation whenever, for all $P R Q$ and $\alpha$:
  - If $P \xrightarrow{\alpha} P'$ then, for some $Q'$, $Q \xrightarrow{\alpha} Q'$ and $P' R Q'$.
  - If $Q \xrightarrow{\alpha} Q'$ then, for some $P'$, $P \xrightarrow{\alpha} P'$ and $P' R Q'$.
- Bisimilarity, written $P \sim Q$, is the largest bisimulation.
- Bisimilarity comes with a proof technique: establish a bisimulation.
Technical problem: stratified bisimulation is not a fixed point.
More importantly, missed an important proof technique.
Bisimulation has impact far outside concurrency.
  - Coinduction and coinductive datatypes (e.g. streams).
  - Applicative bisimulation for $\lambda$-calculi.
  - Coinduction in mechanized proof systems (e.g. Coq and Agda).
  - Non-wellfounded set theory.
Naming

Robin Milner, 1934-2010

All Rights Reserved © 2011 Alcatel-Lucent
In '79, I was talking to Mogens Nielsen and we were really trying to make the pi-calculus work at that time.

It took us [MPW] about three years from about '86, '87, to '88, '89 to get it straight.
Old problem: gensym.

- In the Calculus of Communicating Systems, channel scope is static.
- Cannot model systems with *link mobility*, where channels escape.
- Causes problems modelling functions, objects, references...
Old problem: gensym.

Nielsen and Engberg (1986) *Extended CCS*.

- Name generation modelled by $\alpha$-equivalence.
- Scope extrusion: $P \mid \nu(x)Q \sim \nu(x)(P \mid Q)$ when $x \not\in \text{fn}(P)$.
- Complex: names, variables, constants.
Communicating and Mobile Systems

- Old problem: gensym.
- Nielsen and Engberg (1986) *Extended CCS*.
  - Reduction semantics up to a structural congruence:

\[
P \equiv Q \rightarrow Q' \equiv P' \\
\hline
P \rightarrow P'
\]
Communicating and Mobile Systems

- Old problem: gensym.
- Nielsen and Engberg (1986) *Extended CCS*.
  - In 1989, an LTS semantics, with scope extrusion as a bisimilarity.
  - In 1992, a reduction semantics up to a structural congruence.
  - Scope extrusion and $\alpha$-equivalence *in the structural congruence*.
  - Natural embedding of a $\lambda$-calculus with explicit substitutions.
Old problem: gensym.

Nielsen and Engberg (1986) *Extended CCS*.


Wide applicability of $\pi$-like techniques:

- Business processes.
- Distributed systems.
- Molecular biology.
- Nominal set theory.
- Objects.
- References.
- Security.
- ...
Thank you Robin!