Origins of bisimulation

1. Concurrency [Hennessy, Milner, Park]

   Language equivalence is not appropriate for “interacting automata” because it is not a congruence (w.r.t. natural operations)

2. Modal logic [van Benthem]

   Modal logic is a fragment of first-order logic (over transition graphs). Is there an independent characterisation of which fragment? Bisimulation invariance is key notion.

   $\Phi(x)$ is bisim invariant iff if $T \models \Phi[s]$ and $s \sim t$ then $T \models \Phi[t]$

Generalizing

- Modal μ-calculus (modal logic + least and greatest fixed points) is "mother-of-all-temporal-logics", contains LTL, CTL, CTL*, PDL, ...

- Modal μ-calculus is a fragment of monadic second-order logic (over transition graphs). Which fragment?

- Answer again is the bisimulation invariant fragment (Janin and Walukiewicz, 1996)

$\Phi(x)$ is bisim invariant iff if $T \models \Phi[s]$ and $s \sim t$ then $T \models \Phi[t]$

PhD Problem 1

- **Satisfiability Problem:** "Given a modal μ-calculus formula Φ, is Φ satisfiable?" is **EXPTIME**-complete.

- **Tree Model Property** If Φ is satisfiable then Φ is satisfiable in a transition system/Kripke model that is a (regular) infinite tree.

- **Finite Model Property** If Φ is satisfiable then Φ is satisfiable in a finite transition system/Kripke model.

- **Model Checking Problem** "Given a finite model, a state E and closed Φ, does $E \models \Phi$?" Its exact complexity is a long standing OPEN problem.
  Best known upper bound is **NP** $\cap$ **co-NP**

PhD Problem 2

1. Consider the restricted process language where I is finite

$$E ::= P \mid \sum\{a_i.E_i \,:\, i \in I\} \mid E_1 \mid E_2 \mid E\backslash\{a\}$$

A (closed) process, a finite family $\{P_i \stackrel{\text{def}}{=} E_i \,:\, 1 \leq i \leq n\}$ of definitions, where all the process names in each E_i belong to the set $\{P_1, \ldots, P_n\}$.

"Turing powerful" (simulate Turing machines)

2. However if we disallow restriction

$$E ::= P \mid \sum\{a_i.E_i \,:\, i \in I\} \mid E_1 \mid E_2$$

Bisimulation equivalence is decidable for this fragment. OPEN QUESTION: is observable bisimulation equivalence decidable?

Hardest PhD Problem

Contrasts between language equivalence and bisimulation equivalence

For finite state transition graphs of size n

- Deciding language equivalence is PSPACE-complete. (If the alphabet is singleton then the problem is co-NP complete.)

- Deciding bisimulation equivalence is P-complete. (Naive algorithm is $O(n^2)$)

Show language equivalence isnt definable within first-order logic with fixed points unlike bisimulation equivalence.

Language Theory and Infinite Graphs

- Usual to think of grammars/automata as word generators
- Alternative process calculus view: generators of labelled graphs
- Is this merely cosmetic?
- It allows one to consider other questions.

Pushdown automata

A pushdown automaton, PDA, consists of

- A finite set of states P
- A finite set of stack symbols S
- A finite alphabet A
- A finite set of basic transitions T

Basic transition $pX \xrightarrow{a} q\alpha$ where p, $q \in \mathsf{P}$, $X \in \mathsf{S}$, $a \in \mathsf{A} \cup \{\epsilon\}$ and $\alpha \in \mathsf{S}^*$

Configurations

- A configuration $p\beta$, $p \in \mathsf{P}$ and $\beta \in \mathsf{S}^*$

- Transitions of a configuration

  If $pX \xrightarrow{a} q\alpha \in \mathsf{T}$ then $pX\delta \xrightarrow{a} q\alpha\delta$

Transition graphs $\mathsf{G}(p\beta)$

$\mathsf{P} = \{p, q, r\}$, $\mathsf{S} = \{X\}$, $\mathsf{A} = \{a, b\}$ and T is

$$pX \xrightarrow{a} pXX \qquad pX \xrightarrow{b} r\epsilon \qquad rX \xrightarrow{\epsilon} r\epsilon$$
$$pX \xrightarrow{b} q\epsilon \qquad qX \xrightarrow{b} q\epsilon$$

$\mathsf{G}(pX)$ is

$$\begin{array}{ccccccc}
q\epsilon & \xleftarrow{b} & qX & \xleftarrow{b} & qXX & \xleftarrow{b} & \ldots \\
\uparrow b & & \uparrow b & & \uparrow b & & \\
pX & \xrightarrow{a} & pXX & \xrightarrow{a} & pXXX & \xrightarrow{a} & \ldots \\
\downarrow b & & \downarrow b & & \downarrow b & & \\
r\epsilon & \xleftarrow{\epsilon} & rX & \xleftarrow{\epsilon} & rXX & \xleftarrow{\epsilon} & \ldots
\end{array}$$

Some results

- Language equivalence is undecidable for context-free grammars (and therefore for pushdown automata)

- Bisimulation equivalence is decidable for pushdown automata

- Pushdown automata are richer than context-free grammars with respect to bisimulation equivalence

Other Questions I

- Given a transition graph and vertex α, can one decide what are the reachable vertices ?

$$\{\beta : \alpha \xrightarrow{w} \beta \text{ for some word } w\}$$

- **Proposition:** The set of reachable vertices of a pushdown automaton is regular

- Very useful for infinite-state model checking ("Regular model-checking").

- Active research area: more general kinds of graph

Richer Graphs: Schemes

- A scheme is a finite family

  $$F_i\, x_1^i \ldots x_{n_i}^i \stackrel{\mathrm{def}}{=} t_i,\ 1 \leq i \leq m$$

  of definitions where each F_i is typed and distinct, and each t_i is of based type and is built from the typed variables, $x_1^i, \ldots, x_{n_i}^i$, basis functions, the F_i's and application

- There is also a start configuration S of base type without free variables.

- The order of a scheme is the highest order of a free variable x_j^i that occurs within a definition

2nd-Order Example

$$
\begin{aligned}
F\,x_1x_2x_3 &\stackrel{\text{def}}{=} f(F(Gx_1)(Hx_2)x_3)x_1(x_2(x_3))\\
G\,x_1x_2 &\stackrel{\text{def}}{=} g(x_1(x_2))\\
Hx_1x_2 &\stackrel{\text{def}}{=} h(x_1(x_2))
\end{aligned}
$$

with starting configuration $Fgha$.

More PhD Problems

- Higher-order schema problem: is following decidable? (open for 30 years)

  Given two schemes do they generate the same (infinite) tree?

- The equivalence problem for first-order schemes interreducible to the equivalence checking problem of determinisitic pushdown automata.

- Model-checking problem for schemes: given a scheme, does the tree generated by it have a decidable monadic second-order theory? Decidable Ong (2006) using games