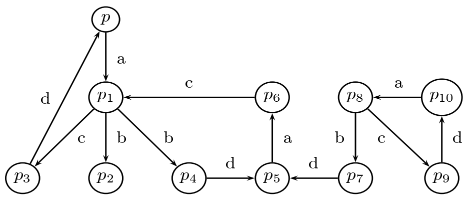
An Introduction to Bisimulation Theory

1. Labelled Transition Systems
2. Simulations and Bisimulations
3. Inductive Characterisation of Strong Bisimulation Equivalence
4. Using Modal Logic to distinguish processes
5. A Language for Processes (CCS)
6. Weak Bisimulation Equivalence
7. Algebraic theories

A Labelled Transition System

States $\{p, p_1, \ldots, p_{10}\}$

Actions: $\{a, b, c, d\}$



$p_1 \xrightarrow{b} p_4$ means that system may move from state p_1 to state p_4 under the (external) stimulus b.

Labelled Transition Systems

An *lts* is a triple $\langle Q, \mathsf{Act}, \mathcal{T}, \rangle$ where

- Q is a set of states
- Act is a set of actions
- $\mathcal{T}$ is a subset of $Q \times \mathsf{Act} \times Q$

(q, a, q') is usually written $q \xrightarrow{a} q'$

Intuition: Elements of Act are *actions* or responses to external stimuli.

Simulations

$S \subseteq Q \times Q$ is a simulation if

$$\begin{tikzcd} P \arrow[d, "a"'] & S & Q & & & P & S & Q \arrow[d, "a"] \\ P' & & & \text{implies} & & P' & S & Q' \end{tikzcd}$$

Every action and choice of P can be matched by Q.

Strong Bisimulation Equivalence

S is a (strong) bisimulation if

- S is a simulation
- S^{-1} is also simulation

Strong Bisimulation Equivalence $P \sim Q$ if there is a bisimulation R such that $P \; R \; Q$.

Thm:

- $\sim$ is an equivalence relation
- $\sim$ is a bisimulation
- $\sim$ is the largest bisimulation

Inductive Characterisation of $\sim$

For $R \subseteq Q \times Q$ let $\mathcal{F}\,(R) \subseteq Q \times Q$ be defined by:

$(p, q) \in \mathcal{F}\,(R) \subseteq Q \times Q$ if

- $p \xrightarrow{a} p'$ implies there is some q' such that $q \xrightarrow{a} q'$ and $(p', q') \in R$
- $q \xrightarrow{a} q'$ implies there is some p' such that $p \xrightarrow{a} p'$ and $(p', q') \in R$

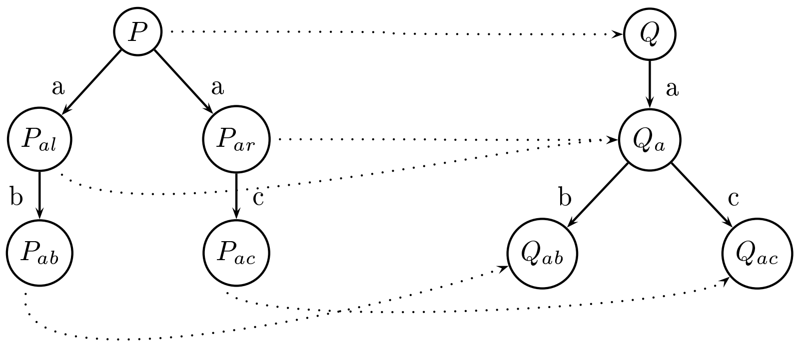
$\mathcal{F}$ is a formal statement of the commuting diagram used to define bisimulations.

Thm:

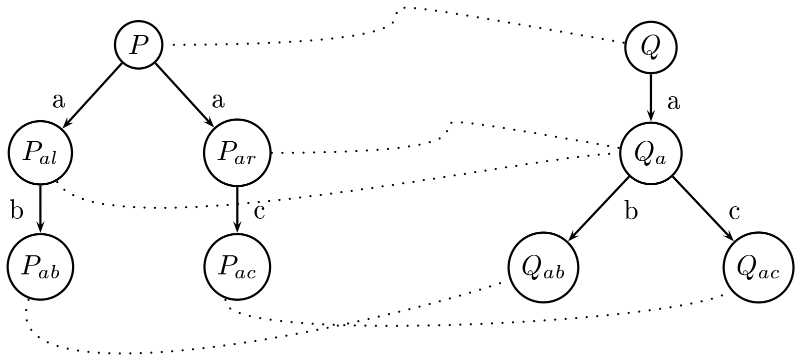
- $\sim$ is the maximal relation such that $\sim \subset \mathcal{F}\,(\sim)$
- $\sim$ is the maximal relation such that $\sim = \mathcal{F}\,(\sim)$

A Simulation

Q can *simulate* P:



Not a Bisimulation



Inductive Characterisation of $\sim$

$$\begin{array}{rcl} \sim_0 & = & Q \times Q \\ \sim_1 & = & \mathcal{F}\,(\sim_0) \\ \ldots\sim_{n+1} & = & \mathcal{F}\,(\sim_n) \\ \ldots\sim_\omega & = & \cap_n\{\sim_n\} \end{array}$$

A decreasing sequence:

$$\sim_0 \supseteq \sim_1 \supseteq \sim_2 \ldots \supseteq \sim_n \supseteq \sim_{n+1} \ldots\ldots \sim_\omega \qquad\qquad \supseteq \sim$$

Does $\sim_\omega$ ever reach $\sim$?

Finite-Branching ltss

An lts is **finite-branching** if for every state q and every action a,

$$\{\, q' \mid q \xrightarrow{a} q' \,\} \quad \text{is finite}$$

Thm: In a finite branching lts $\sim$ coincides with $\sim_\omega$.

Not true in general.

Using Modal Logic_{to distinguish processes}

Language for properties:

$$\phi \quad ::= \quad \mathbf{true} \mid \neg\phi \mid \langle a \rangle \phi$$

Interpretation:

- $p \vdash \mathbf{true}$ for every p
- $p \vdash \neg\phi$ if $p \not\vdash \phi$
- $p \vdash \langle a \rangle \phi'$ if $p \xrightarrow{a} p'$ where $p' \vdash \phi'$

Shorthand:

- $\mathbf{false} = \neg\mathbf{true}$
- $[a]\phi = \neg\langle a \rangle(\neg\phi)$

Logical Equivalence

Properties enyoyed by a process: $\mathcal{L}(p) = \{\, \phi \mid p \vdash \phi \,\}$

Property Equivalence: $p \sim_{\mathcal{L}} q$ if $\mathcal{L}(p) = \mathcal{L}(q)$

Thm: $p \sim_{\mathcal{L}} q$ if and ony if $p \sim_\omega q$

Consequence: In a finite branching lts $\sim_{\mathcal{L}}$ coincides with $\sim$

Consequence: If $p \not\sim q$ then there is some ϕ such that $p \vdash \phi$ and $p \not\vdash \phi$

Examples

Semaphores:

$$
\begin{aligned}
S_1 &\Leftarrow p?v!S_1 \\
S_2 &\Leftarrow p?S_2^1 \\
S_2^1 &\Leftarrow p?S_2^2 + v!S_2 \\
S_2^2 &\Leftarrow v!S_2^1
\end{aligned}
$$

Buffers:

$$
\begin{aligned}
B(i,o) &\Leftarrow i?o!B\langle i,o\rangle \\
TB_{io} &\Leftarrow (\mathsf{new}\, c)\ (B\langle i,c\rangle \mid B\langle c,o\rangle)
\end{aligned}
$$

Examples

Semaphores:

$$S_1 \mid S_2 \sim S_2$$

Buffers:

$$TB_{io} \sim \mathrm{SC}$$

where

$$\mathrm{SC} \Leftarrow i?\tau o!\mathrm{SC}$$

But

$$T_{io} \not\sim \mathrm{C}$$

where

$$\mathrm{C} \Leftarrow i?o!\mathrm{C}$$

A Language for Processes

$$
\begin{aligned}
P &::= \mathbf{0} \mid \mu P \mid P + P \\
&\quad\ \mid P \mid P \mid (\mathsf{new}\, c)\ P \mid A\langle \tilde{c}\rangle \\
\mu &::= c? \mid c! \mid \tau \qquad c \in \mathit{Chans}
\end{aligned}
$$

Each constant A has a definition associated with it:

$$A(\tilde{c}) \Leftarrow D_A$$

LTSs for process terms

Action:

$$\mu P \xrightarrow{\mu} P \qquad \frac{P \xrightarrow{\alpha} P',\ Q \xrightarrow{\overline{\alpha}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

Structure:

$$\frac{P \xrightarrow{\mu} P'}{(\mathsf{new}\, c)\ P \xrightarrow{\mu} (\mathsf{new}\, c)\ P'}\ c \notin \mu \qquad \frac{D_A\{\!|\tilde{a}/\tilde{c}|\!\} \xrightarrow{\mu} P'}{A\langle \tilde{a}\rangle \xrightarrow{\mu} P'}$$

$$\frac{P \xrightarrow{\mu} P'}{\begin{array}{l} P + Q \xrightarrow{\mu} P' \\ P \mid Q \xrightarrow{\mu} P' \mid Q \end{array}}$$

Weak Bisimulation Equivalence

S is a *weak bisimulation* if

- S is a weak simulation
- S^{-1} is also weak simulation

Weak Bisimulation Equivalence $P \approx Q$ if there is a bisimulation R such that $P\ R\ Q$.

Thm:

- $\approx$ is an equivalence relation
- $\approx$ is a weak bisimulation
- $\approx$ is the largest weak bisimulation

Implementation

Parametric Task:

$$\begin{array}{lcl} T(a,b,c,d) & \Leftarrow & a?C \\ C & \Leftarrow & c!E \\ E & \Leftarrow & b!D + d?B \\ D & \Leftarrow & d?T \\ B & \Leftarrow & b! \end{array}$$

$$\begin{array}{lcl} \mathrm{Sys}_n & \Leftarrow & (\mathsf{new}\, c_1, \ldots, c_n)\ A\langle a_1, b_1, c_1, c_{n-1}\rangle \mid \\ & & \qquad D\langle a_2, b_2, c_2, c_1\rangle \mid \ldots \mid D\langle a_n, b_n, c_n, c_{n-1}\rangle \end{array}$$

Result: $\mathrm{Spec}_n \approx \mathrm{Sys}_n$

Weak Bisimulations

$S \subseteq Q \times Q$ is a *weak simulation* if

$$\begin{array}{ccccccccc} P & S & Q & & & P & S & Q \\ s\Big\Downarrow & & & & \text{implies} & & & s\Big\Downarrow \\ P' & & & & & P' & S & Q' \end{array}$$

Every **sequence** of actions and of P can be matched by Q.

$P \xRightarrow{a_1 \ldots a_n} Q$ means:

$$P \xrightarrow{\tau}{}^{*} \xrightarrow{a_1} \xrightarrow{\tau}{}^{*} \ldots \xrightarrow{\tau}{}^{*} \xrightarrow{a_n} \xrightarrow{\tau}{}^{*} Q$$

Milners Scheduler

Background: n tasks. T_i begins with a_i and ends with b_i. T_{i+1} (mod n) can not start before T_i.

Requirements:

- $b_i!$ can not happen before $a_i?$.
- $a_{i+1}?$ can not happen before $a_i?$

Specification: (+ is mod n)

$$\begin{array}{rcll} \mathrm{Spec}_n & \Leftarrow & \mathrm{Sched}_{1,\emptyset} & \\ \mathrm{Sched}_{i,R} & \Leftarrow & \sum_{j \in R} b_j!\mathrm{Sched}_{i,R-\{j\}}, & \text{if } i \in R \\ & & \sum_{j \in R} b_j!\mathrm{Sched}_{i,R-\{j\}} + a_i?\mathrm{Sched}_{i+1,R\cup\{i\}}, & \text{otherwise} \end{array}$$

$\mathrm{Sched}_{i,R}$ is state in which T_i can start and R is set of running tasks.