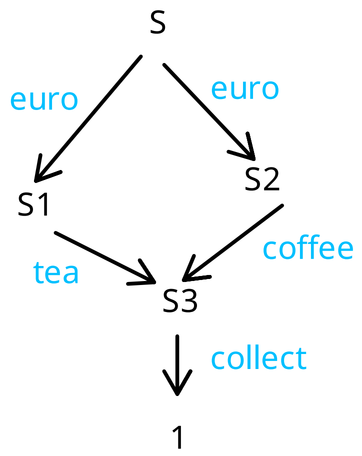
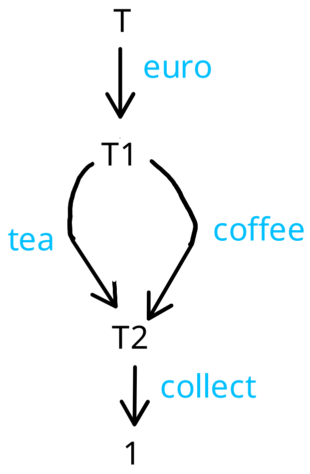
Equivalences of concurrent programs

S = (euro.tea + euro.coffee).collect



T = euro.(tea + coffee).collect



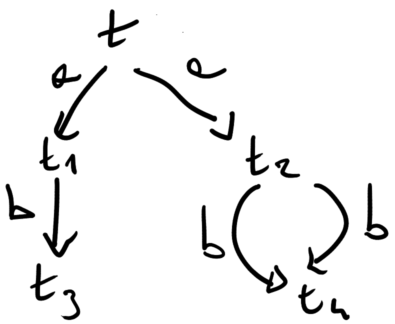
- S and T have the same traces (words), but they differ if interpreted as reactive systems
- For reactive systems, bisimulation is a better notion of equivalence than language (trace) equivalence

Def. Given an LTS T, a binary relation B on the states of T is a bisimulation if whenever q B r
- for all q --a--> q' there is r --a--> r' such that q' B r' and
- for all r --a--> r' there is q --a--> q' such that q' B r'

Exercise 11
Let S and T as in the example of the vending machine above. Show that there is no bisimulation containing the pair (S,T).

Odd cases

Bisim. ?



$$B = \{ (s,t), (s_1,t_1), (s_1,t_2), (t_1,t_2), (s_2,t_3), (s_2,t_4), (t_3,t_4) \}$$

Properties of bisimulations

<u>Thm</u>: Given a transition system on S, Id_S is a bisimulation

Proof: Exercise

<u>Thm</u>: Given a set of indexes I, let B_i be bisimulations for all $i \in I$

(1) B_i^{-1} is a bisimulation for all $i \in I$

(2) $\bigcup_{i \in I} B_i$ is a bisimulation

(3) $B_i \circ B_j$ is a bisimulation for all $i, j \in I$

Proof (1): $(s,t) \in B_i^{-1} \iff (t,s) \in B_i$

for each $t \xrightarrow{a} t'$ there is $s \xrightarrow{a} s'$ s.t. $(s',t') \in B_i \implies (t',s') \in B_i^{-1}$

and likewise for each $s \xrightarrow{a} s'$ using the first clause of Def of bisim.

Proof (2): Exercise

Proof (3) $(r,t) \in B_i \circ B_j \iff \exists s : (r,s) \in B_i$ and $(s,t) \in B_j$

$\Rightarrow \forall\, r \xrightarrow{a} r' \;\exists\, s \xrightarrow{a} s' : r' B_i s'$

$\Rightarrow \exists\, t \xrightarrow{a} t' : t B_j t'$

And likewise for the other clause $\square$

Bisimilarity $\quad \sim \;=\; \bigcup \{ B : B \text{ is a bisimulation} \}$

<u>Thm</u> $\sim$ is the largest bisimulation (!)

<u>Thm</u> $\sim$ is an equivalence relation

What about communication?

Let $A_\perp = A \cup \{\perp\}$ $\quad \perp \notin A$ and fix a communication function

$$\_\circ\_ : A_\perp \times A_\perp \longrightarrow A_\perp \quad \left\{ \begin{array}{l} \circ \text{ commutative} \\ \circ \text{ associative} \\ \forall a \in A_\perp : a \circ \perp = \perp \circ a = \perp \end{array} \right.$$

$$(\mathrm{com}_1)\ \frac{x \xrightarrow{a} x' \quad y \xrightarrow{b} y' \quad a \circ b \in A}{x \parallel y \xrightarrow{a \circ b} x' \parallel y'} \qquad (\mathrm{com}_2)\ \frac{x \xrightarrow{a} 1 \quad y \xrightarrow{b} y' \quad a \circ b \in A}{x \parallel y \xrightarrow{a \circ b} y'}$$

$$(\mathrm{com}_3)\ \frac{x \xrightarrow{a} x' \quad y \xrightarrow{b} 1 \quad a \circ b \in A}{x \parallel y \xrightarrow{a \circ b} x'} \qquad (\mathrm{com}_4)\ \frac{x \xrightarrow{a} 1 \quad y \xrightarrow{b} 1 \quad a \circ b \in A}{x \parallel y \xrightarrow{a \circ b} 1}$$

An instance of the above framework is CCS where synchronisation happens between a "sender" an a "receiver" thread on a port 'a': the alphabet is partitioned in a set In of "input ports" and a set $Out = \{\bar{a} \mid a \in I\}$ of "output ports" and $\bar{a} \circ a = \tau = a \circ \bar{a}$ for all $a \in In$

Example

Show that $ax + by \parallel cz \xrightarrow{b} x \parallel z$ if $a \circ c = b$, $x \neq 1$, and $y \neq 1$

$$\dfrac{\dfrac{\dfrac{\dfrac{a \in A}{a \xrightarrow{a} 1}\ \text{Act}}{ax \xrightarrow{a} x}\ \text{seq1}}{ax + by \xrightarrow{a} x}\ \text{Cho1} \qquad \dfrac{\dfrac{c \in A}{c \xrightarrow{c} 1}\ \text{Act}}{cz \xrightarrow{c} z}\ \text{seq2}}{ax + by \parallel cz \xrightarrow{b} x \parallel z}\ \text{Com1}$$

Summary

- FM: what for & basic (fundamental questions)
- Brief overview of concurrency
  - Problems
  - Shared-memory vs communication
- Operational semantics
  - Transition Systems
  - Structural operational semantics
  - Reg Exp
  - BPAs
- Bisimulations & bisimilarity