Modelling systems

In order to model, we will assume first-order language $\mathcal{F}$ + transition system $\langle V, S, I, T \rangle$

V - finite set of variables → they represent storage elements and program counters pointing to the current location

S - set of states → each state is assigning values to variables from their domain

- if state satisfies a first-order formula φ from $\mathcal{F}$ & it is represented by $\underline{s \models \varphi}$

I - initial condition → unquantified first-order formula of $\mathcal{F}$

- state is initial if $s \models I$

T - finite set of transitions - transition: $\tau = en_\tau \rightarrow f_\tau$

- en_τ: enabling condition (first-order formula from $\mathcal{F}$

- f_τ: $\underbrace{(v_1, v_2, \dots, v_n)}_{\text{vars from } V} := \underbrace{(e_1, e_2, \dots, e_n)}_{\text{expressions of } \mathcal{F}}$

Execution

Showcase of execution of transition system (it can be finite or infinite sequence of states)

$$I: \left.\begin{array}{l} c = a \\ d = b \end{array}\right\} \text{inputs - } a, b \text{ are inputs to variables (initial values)}$$

$\quad e = 0$

enabling condition

T: τ_1: $c > 0$ → $(c, e) := (c-1, e+1)$ - execution will terminate iff $c = 0 \wedge d = 0$

$\quad\tau_2$: $d > 0$ → $(d, e) := (d-1, e+1)$ (no possible transitions after that)

at the initial state both τ_1 and τ_2 are enabled, so a choice exist of which transition to pick
⇒ nondeterministic → (I picked that $a=2 \wedge b=1$ at the initial state)

One possible execution:

$s_0: \langle a=2, b=1, c=2, d=1, e=0 \rangle$

$s_1: \langle a=2, b=1, c=1, d=1, e=1 \rangle$ (I used τ_2)

$s_2: \langle a=2, b=1, c=1, d=0, e=2 \rangle$ (I used τ_2)

$s_3: \langle a=2, b=1, c=0, d=0, e=3 \rangle$ (I used τ_1)