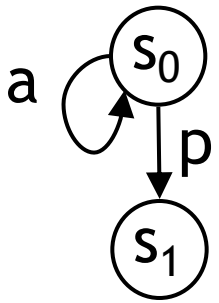
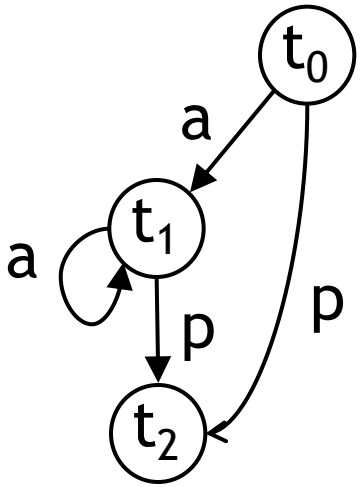
# Exercises:
# Communicating automata
# Timed automata
# Process Algebras (CCS)

# CA: Exercise

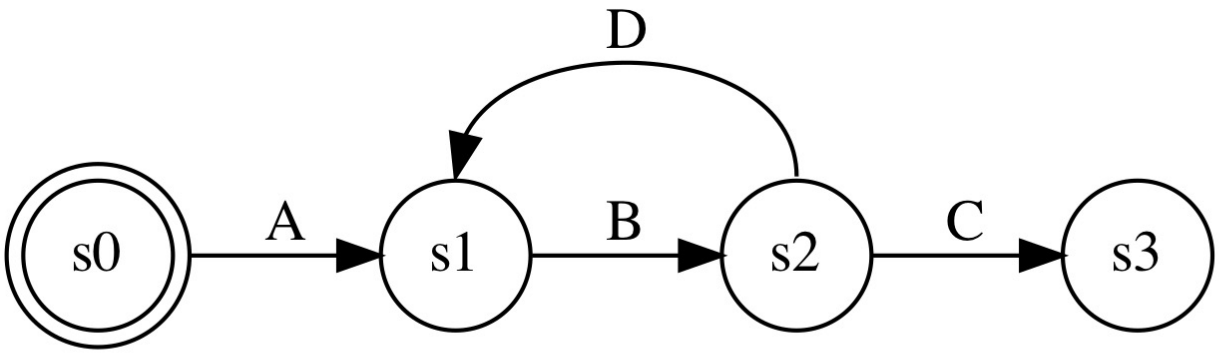
- Are these LTSs bisimilar? ($s_0 \sim t_0$?)

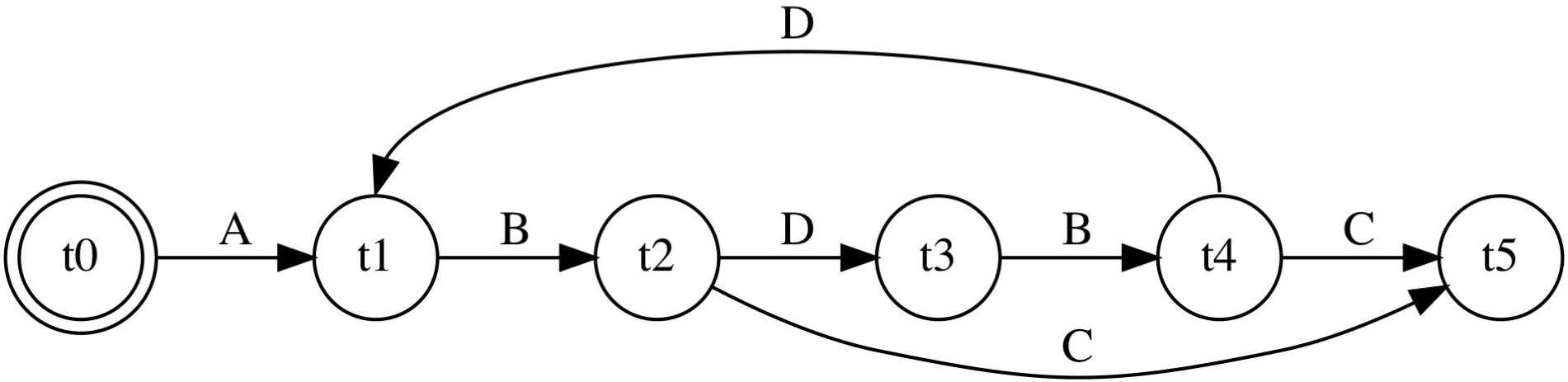


# CA: Exercise

- Are s0 and t0 strongly bisimilar? ($s_0 \sim t_0$ ?)



# TA: Worker and Hammer, part 1

- W starts in state *rest*
  - Initially, the worker is idle
- From *rest*, W can do start! and go to state *work*
  - The worker picks up a hammer and starts working
- From *work*, W can do done! and go to *rest*
  - The worker has done some work and needs to rest

- Time constraints:
  - The worker cannot work for more than 60 TU
  - The worker cannot work for less than 10 TU

# TA: Worker and Hammer, part 2

- H starts in state *free*
  - initially, the hammer is not being used
- From *free*, H can do start? and go to state *busy*
  - A worker picks up the hammer and starts using it
- From *busy*, H can do hit! and go to *busy*
  - The hammer is being used to hit nails
- From *busy*, H can do done? And go to *free*
  - The worker has put the hammer down
- Time constraints:
  - At least 1 TU between two hit! actions
  - done? may only happen at least 5 TU after start?

# Exercise

Describe a vending machine V such that:

- Initially, V accepts a coin, which may be either a nickel or a dime
- After receiving any coin, V makes a cup of coffee or tea
- After making the cup, V gives it to the user
- Then, it terminates
- Actions: *{ nickel, dime, makeC, makeT, giveC, giveT }*

- Write the LTS describing V's behaviour
- Can you write it as a CCS process?

# A brief tour of CAAL (1/2)

You can use CAAL to view the LTS of a CCS process.

Type this into "*Edit*", then go to "*Explore*"

```
agent V1 =
  nickel.('makeC.'giveC.0 + 'makeT.'giveT.0)
+ dime.('makeC.'giveC.0 + 'makeT.'giveT.0) ;
agent V2 = (NickelOrDime | MakeCup) \ {done};
agent NickelOrDime = nickel.'done.0 + dime.'done.0;
agent MakeCup = done.('makeC.'giveC.0 + 'makeT.'giveT.0);
```

- **V1** is $V_{CCS}$; **V2** is $V'_{CCS}$

# A brief tour of CAAL (2/2)

- Syntax:
  - CAAL uses `0` for the **nil** process, `tau` for τ
  - Apostrophe for co-names (e.g., ‘done = $\overline{done}$)
  - Process names: 1st letter must be *Uppercase*
  - Channel names: 1st letter must be *lowercase*
- CAAL can also check (strong/weak) bisimulation
  - Weak bisim. = a variant of branching bisimulation
- Go to “*Verify*”, then “Add property”
  - Select Equivalence/Preorder checking
  - Choose which equivalence to check

# Exercise (1/2)

A computer scientist publishes a paper, then offers a coin, then drinks coffee, then starts over:

$$CS \triangleq \overline{pub}.\overline{coin}.coffee.CS$$

A coffee machine receives a coin, then offers a coffee, then starts over:

$$CM \triangleq coin.\overline{coffee}.CM$$

- Draw the LTS for the system

$$P \triangleq (CM \mid CS) \setminus \{coin, coffee\}$$

- (Is it branching bisimilar to $S \triangleq \overline{pub}.S$ ?)

# Exercise (2/2)

$CS \triangleq \overline{pub}.\overline{coin}.coffee.CS$

$CM \triangleq coin.\overline{coffee}.CM$

LTS of $P \triangleq (CM \mid CS) \setminus \{coin, coffee\}$ ?

- (Is $P$ branching bisimilar to $S \triangleq \overline{pub}.S$ ?)

$$\frac{}{\mu.P \xrightarrow{\mu} P} \qquad \frac{P \xrightarrow{\mu} P' \quad \mu \neq a \quad \mu \neq \bar{a}}{P \setminus a \xrightarrow{\mu} P' \setminus a} \qquad \frac{P \xrightarrow{\mu} P' \quad K \triangleq P}{K \xrightarrow{\mu} P'}$$

$$\frac{P \xrightarrow{\mu} P'}{P \mid Q \xrightarrow{\mu} P' \mid Q} \qquad \frac{Q \xrightarrow{\mu} Q'}{P \mid Q \xrightarrow{\mu} P \mid Q'} \qquad \frac{P \xrightarrow{a} P' \; Q \xrightarrow{\bar{a}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

# Exercise: Barrier synchronization

- Write a CCS process such that
  - It performs 3 actions (say, a b c) in any order (in parallel)
  - After all 3 actions have been performed, it performs another action (say, d) and terminates

- Hint: you will need to introduce new actions and force synchronization on them