Advanced state modeling

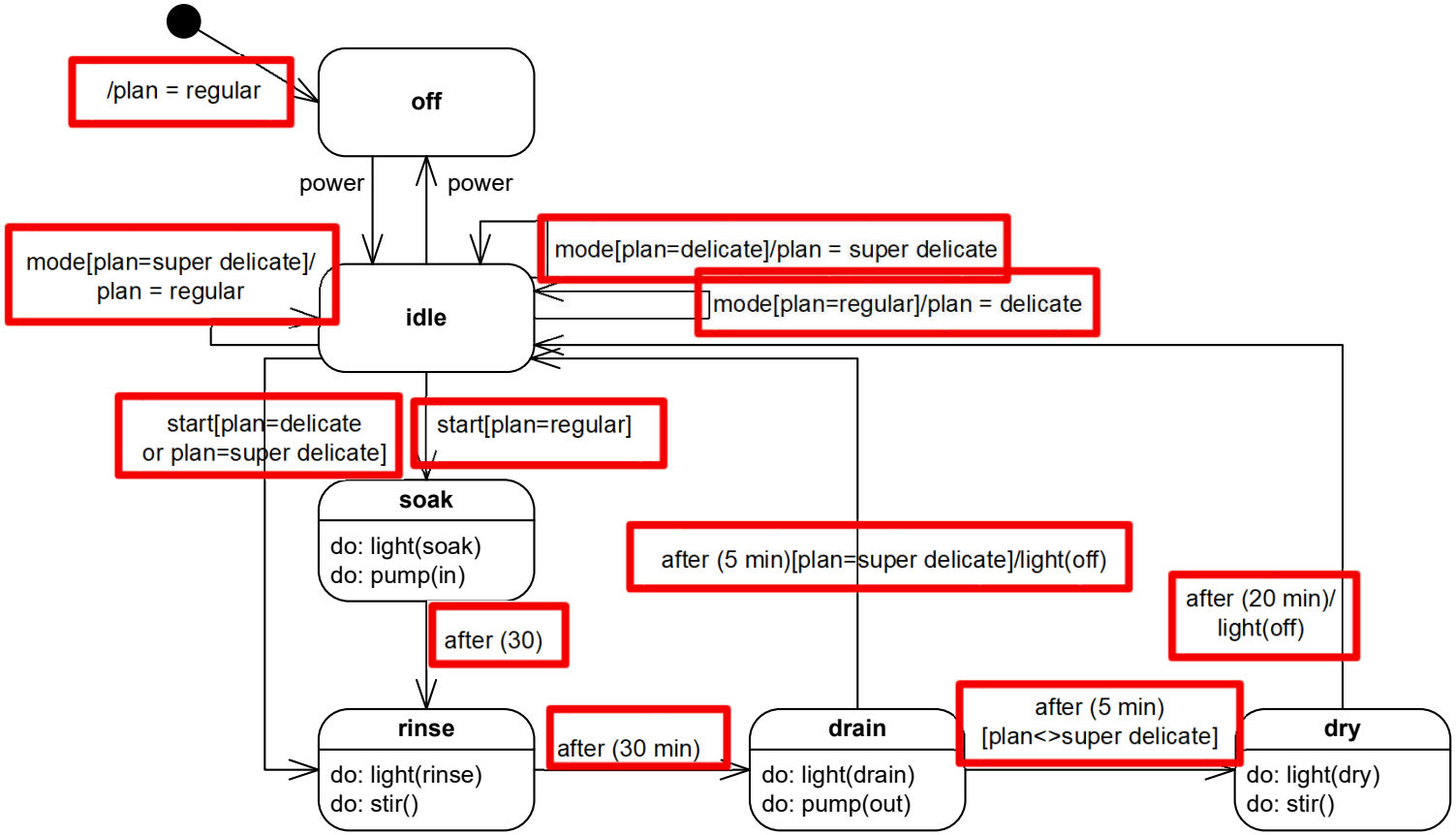
Marlon Dumas

Institute of Computer Science

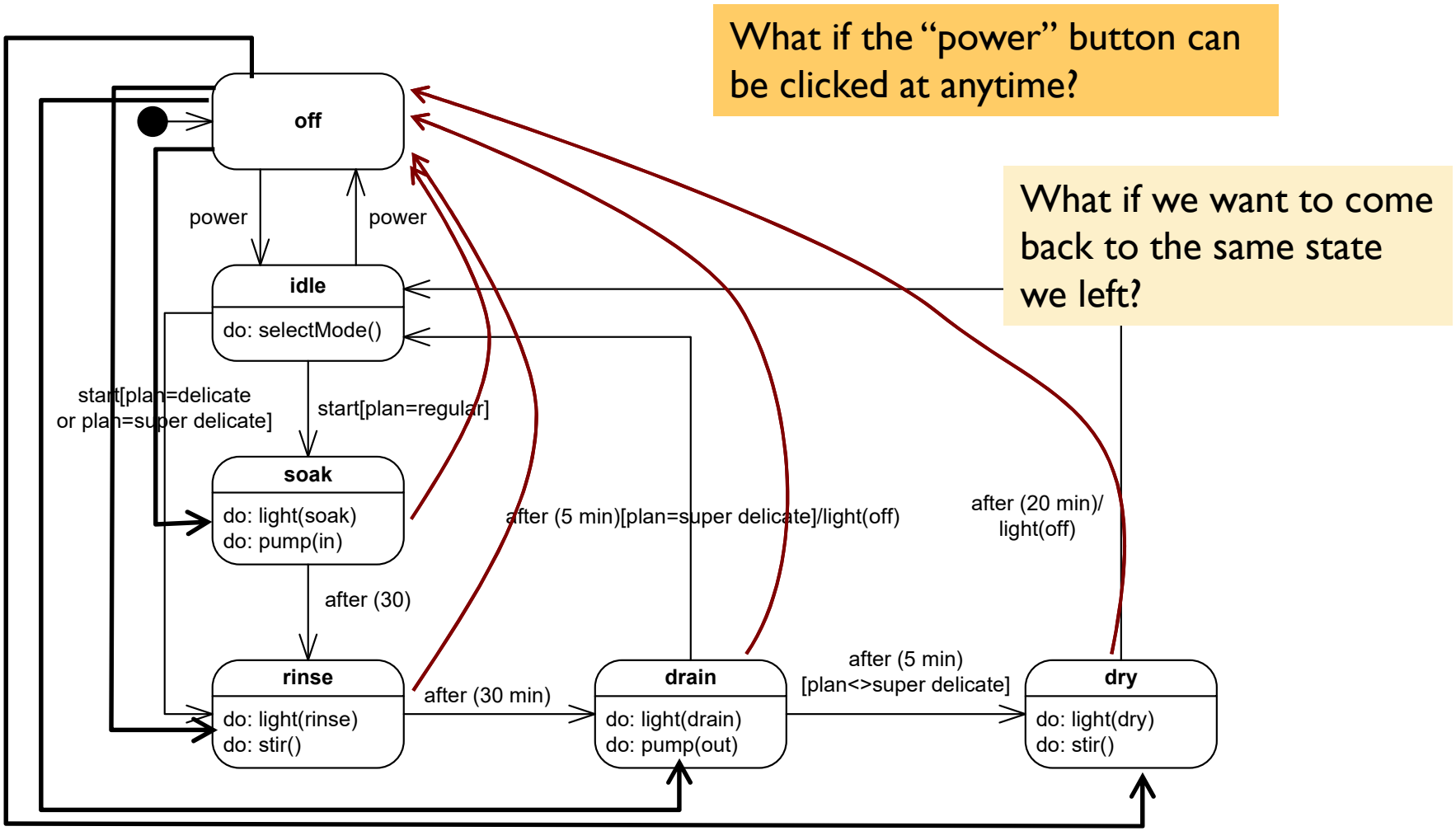
How does a washing machine work?

- On/off (power) button.
- Start button (no stop button!)
- Light indicates current stage
  - soaking, rinsing, draining, drying
- Three washing plans that can be changes using a “mode” button:
  - Regular
  - Delicate (no soaking)
  - Super delicate (no soaking, no drying)
- Off can be pushed only:
  - before starting
  - or after finishing

Statechart for the washing machine

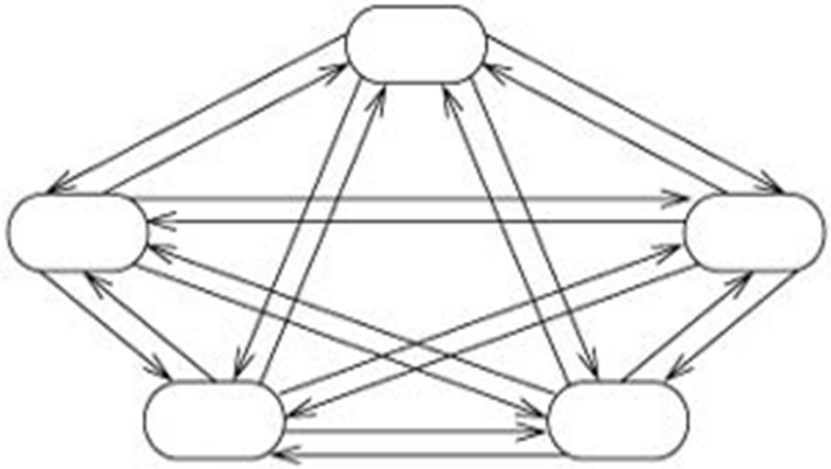


State explosion and history



State explosion

- If we have "n" classes with "m" (boolean) attributes each
  (let's assume that all classes have the same number of attributes)
  - Possible states of the whole system = 2^{nm}



Abstraction in Statecharts

Factor out common behavior

Composite States

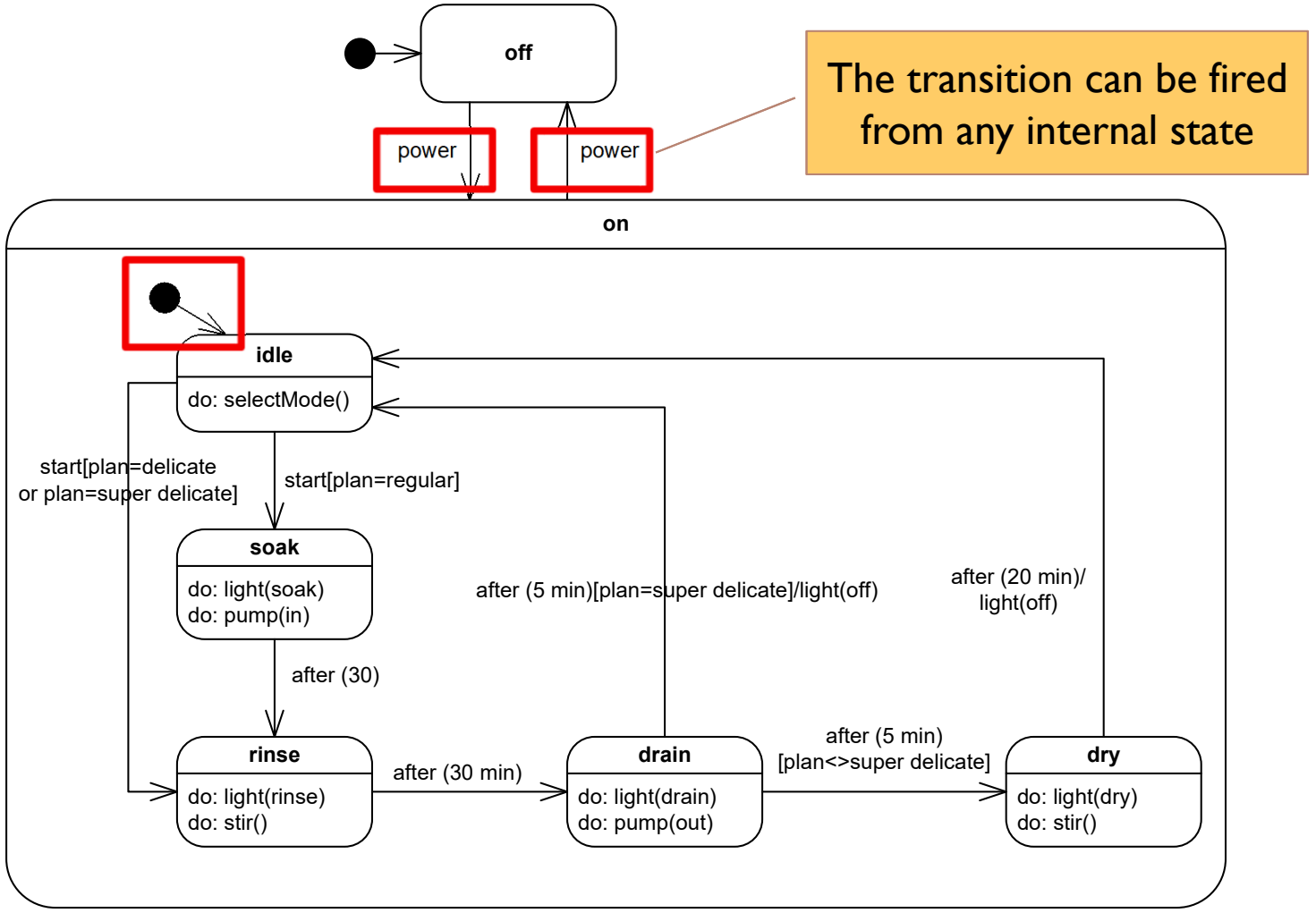
Remember history

History pseudo-states

Segregate independent behavior

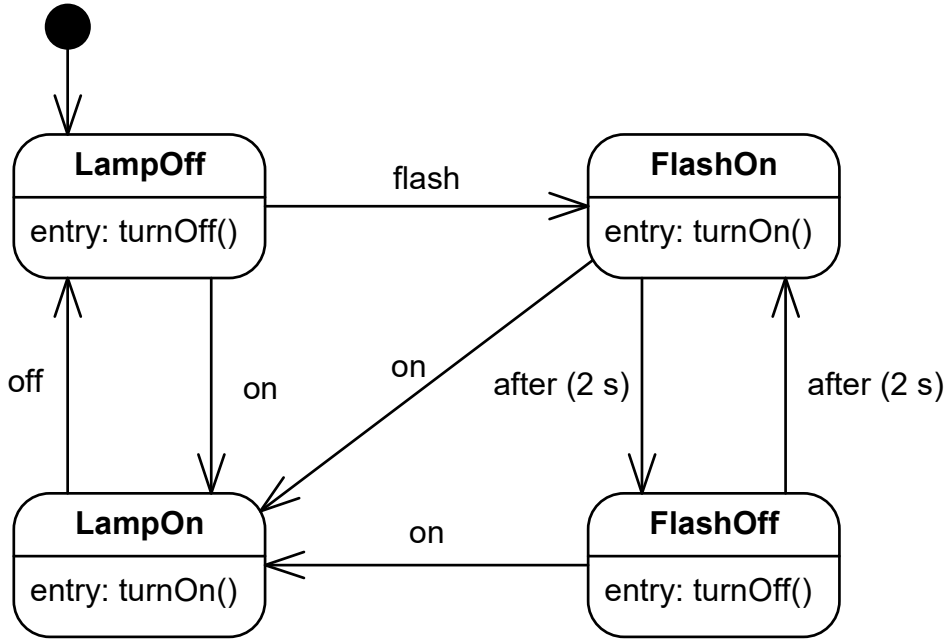
Orthogonal/ Parallel States

Composite states



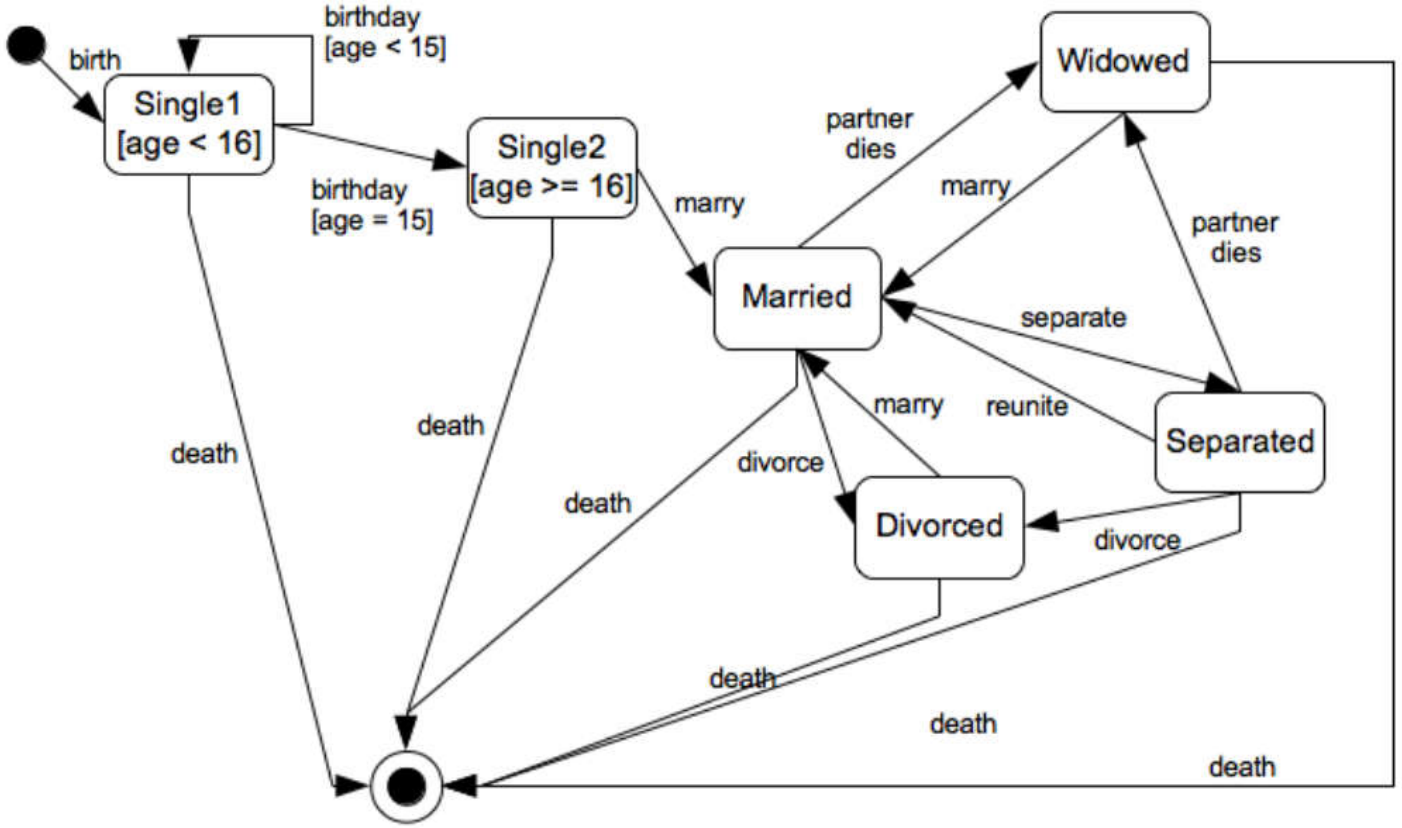
Exercise 1

- Group "FlashOn" and "FlashOff" states into a composite state "Flashing"



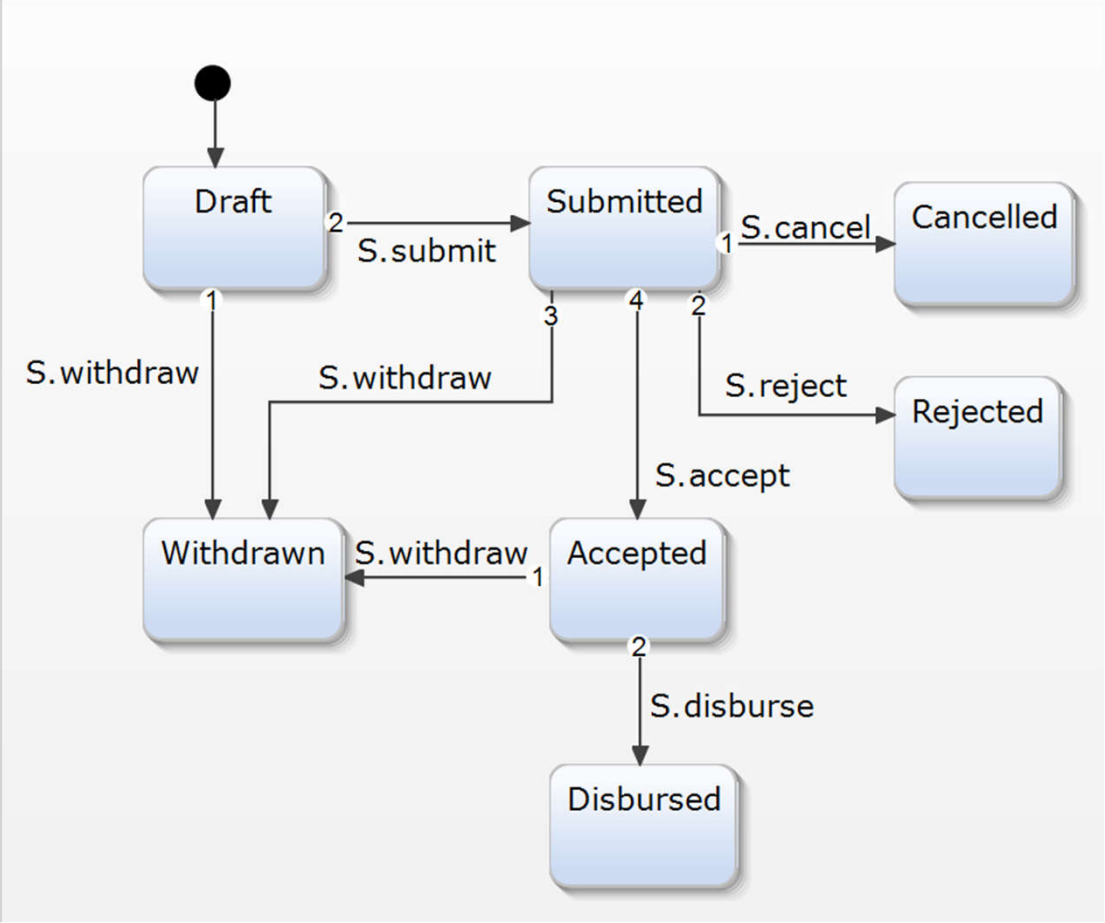
Exercise 2

- Fix and simplify this statechart



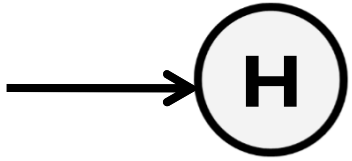
Exercise 3

- Simplify this statechart

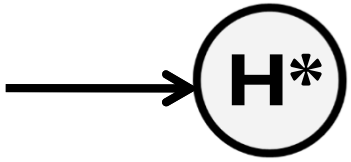


History pseudo-state

- Return to a previously visited hierarchical state
- Shallow history: just the current level

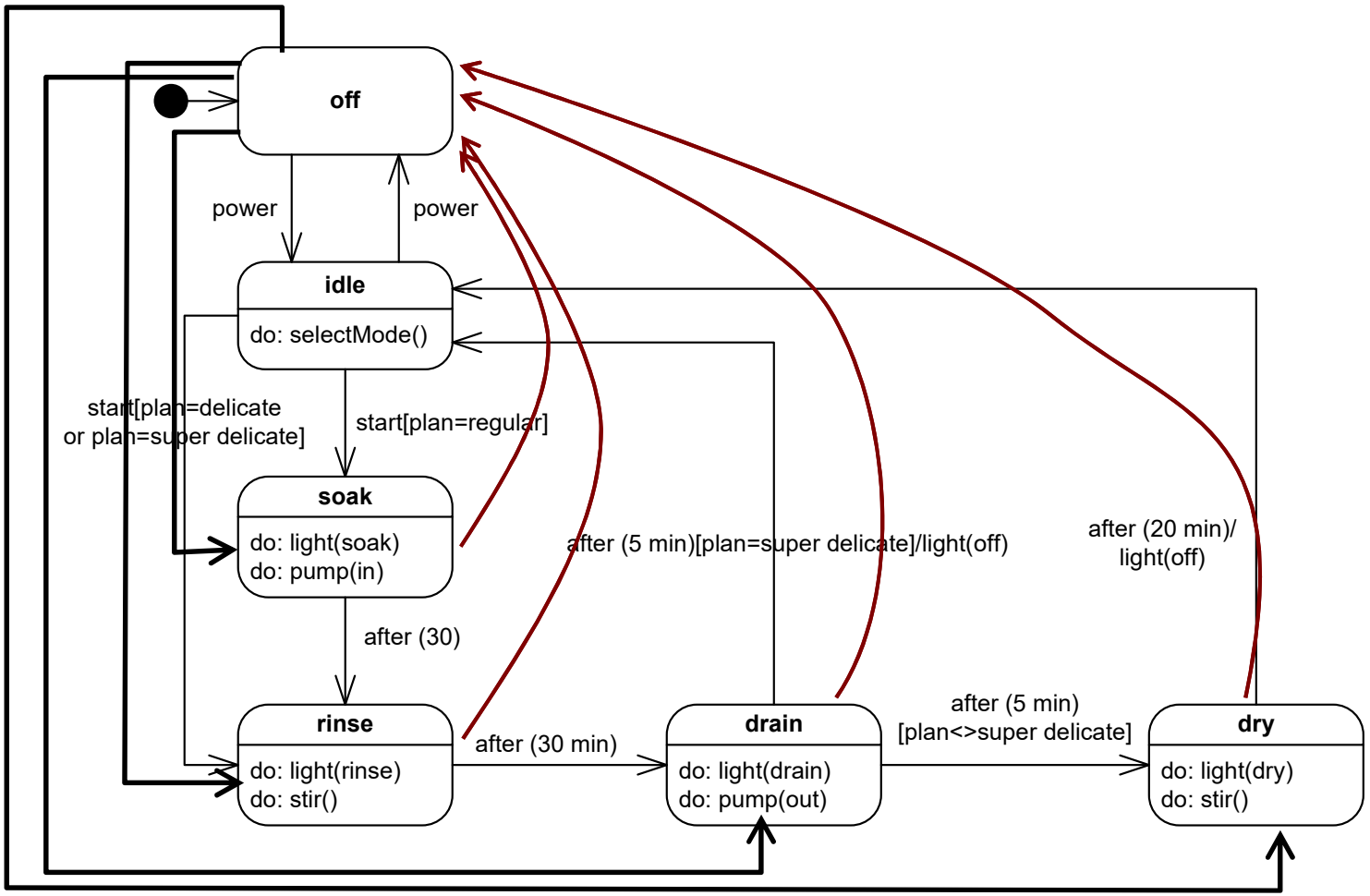


- Deep history: includes all nested states

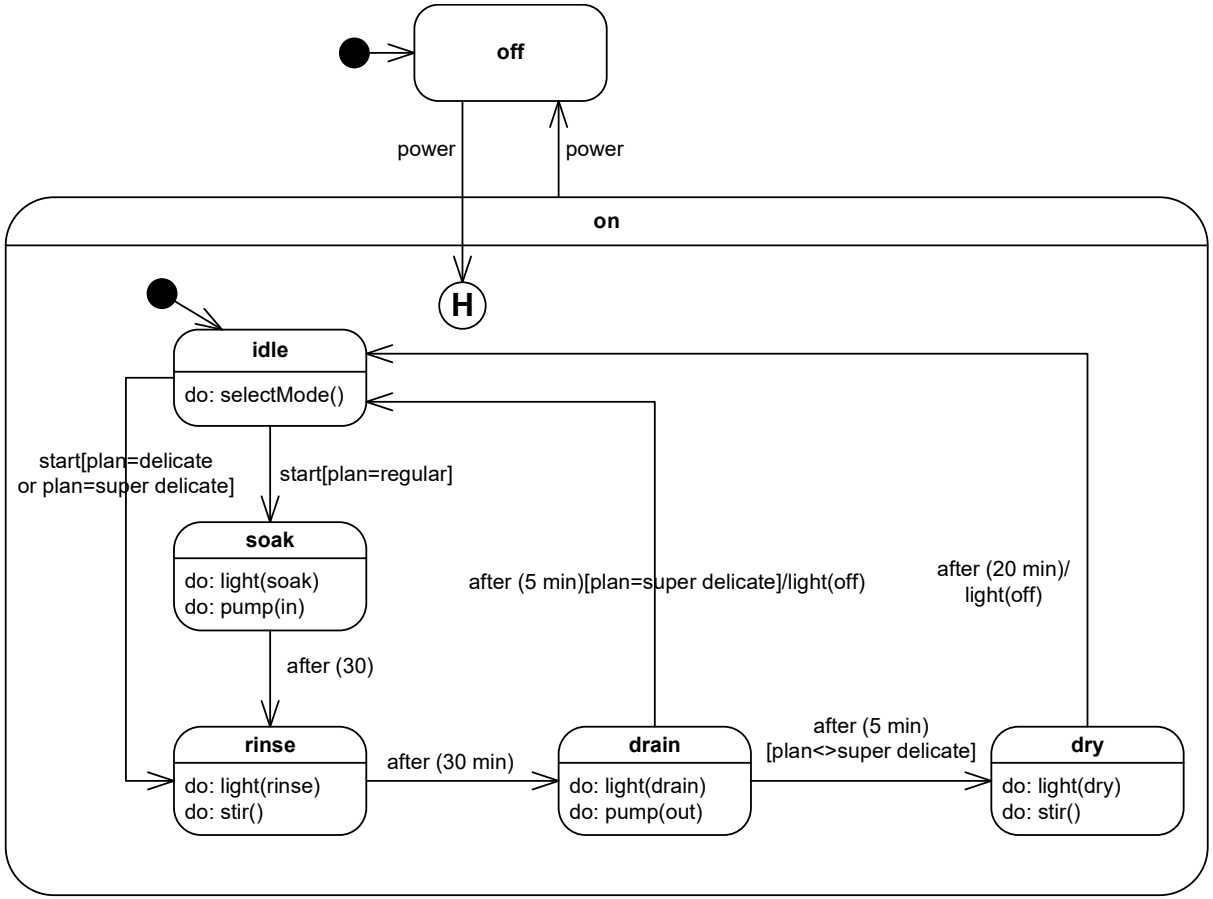


- Sometimes it is useful to clear history:
  - clear-history(state) clh(state)
  - clear-history(state*) clh(state*)

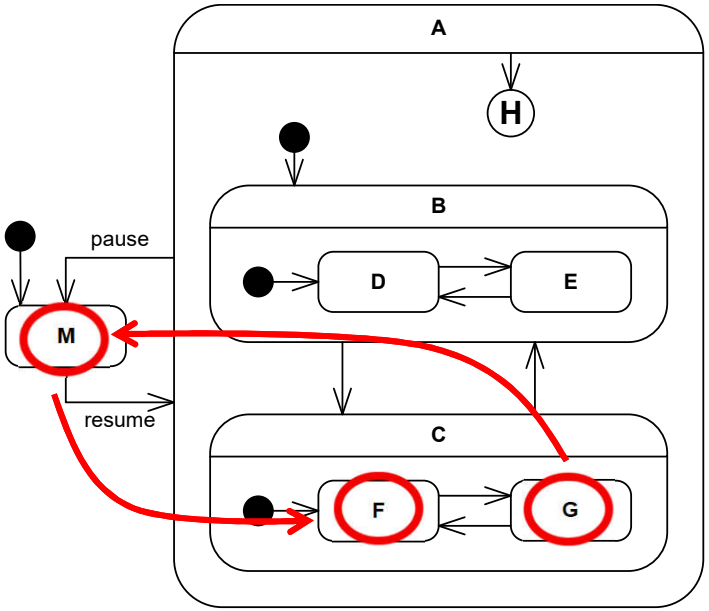
Back to the washing machine…

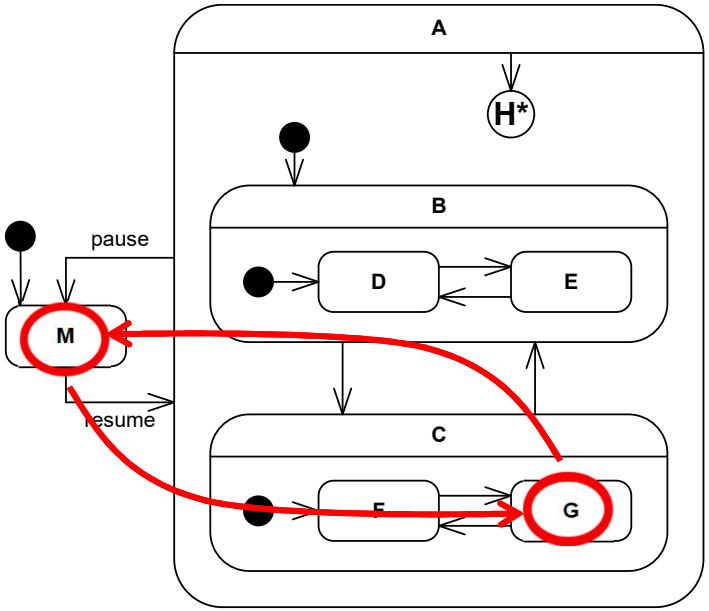


Washing machine with “history”



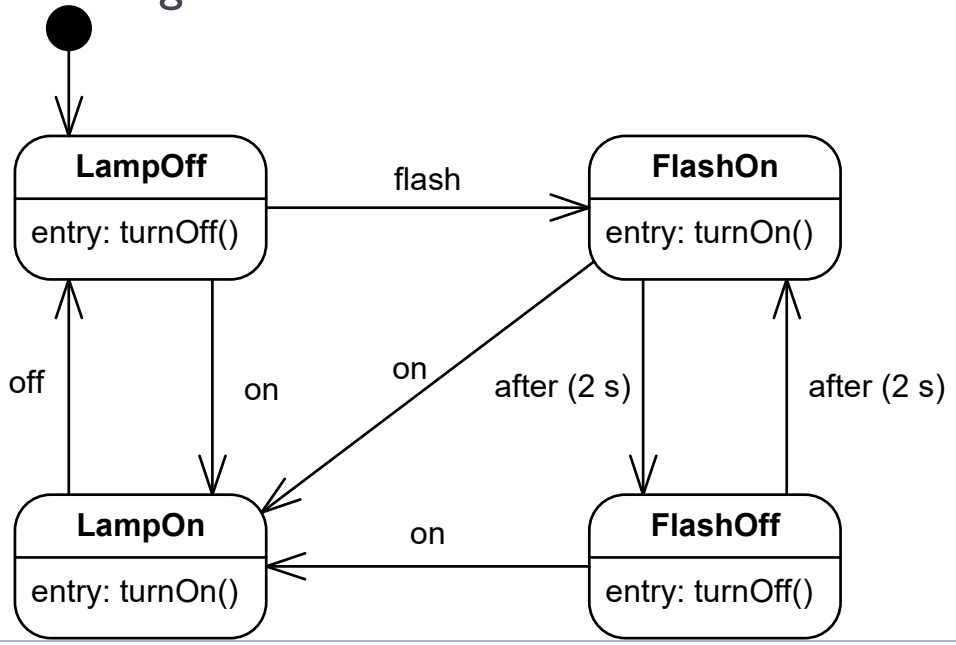
Shallow vs. Deep history





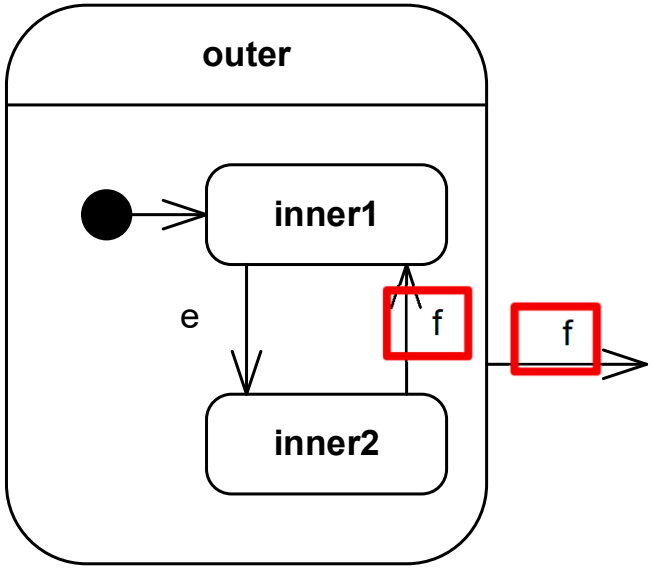
Exercise 4

- Revise the follow statechart such that:
  - There is no flash button. The only way to make the lamp flash is to push the "on" button when we are in the LampOn state
  - To stop flashing, we must push on the "on" button or the "off" button
  - If we push "off" while the lamp is flashing, and then we push "on", the lamp re-starts flashing



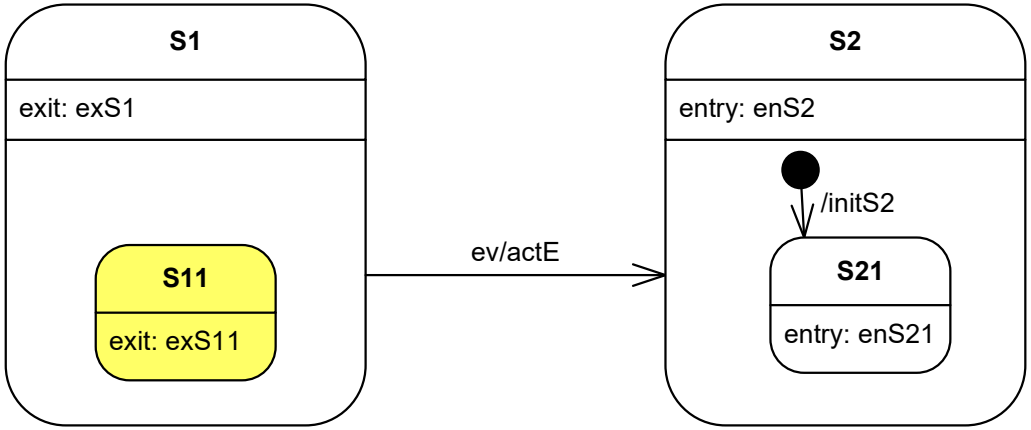
Note on transition precedence

- Two or more transitions may have the same event trigger
  - inner transition takes precedence
  - if no transition is triggered, event is discarded



Order of activities in nested models

- Same approach as for the simple case

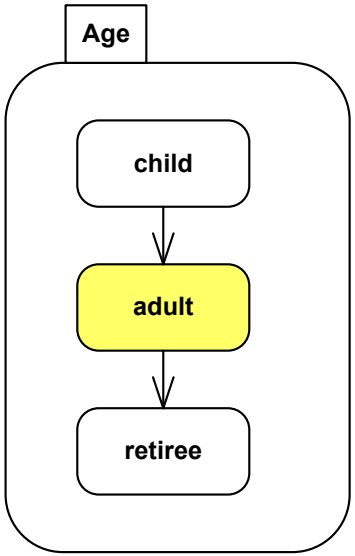


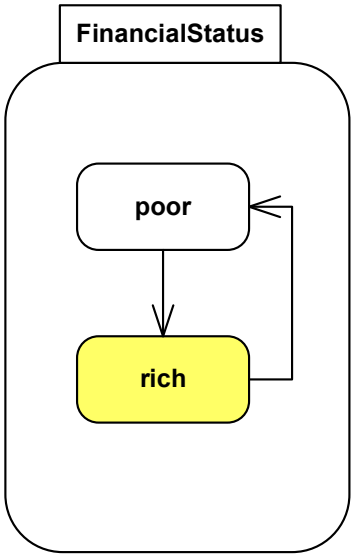
Execution sequence:

exS11 ⇨ exS1 ⇨ actE ⇨ enS2 ⇨ initS2 ⇨ enS21

Independent behavior

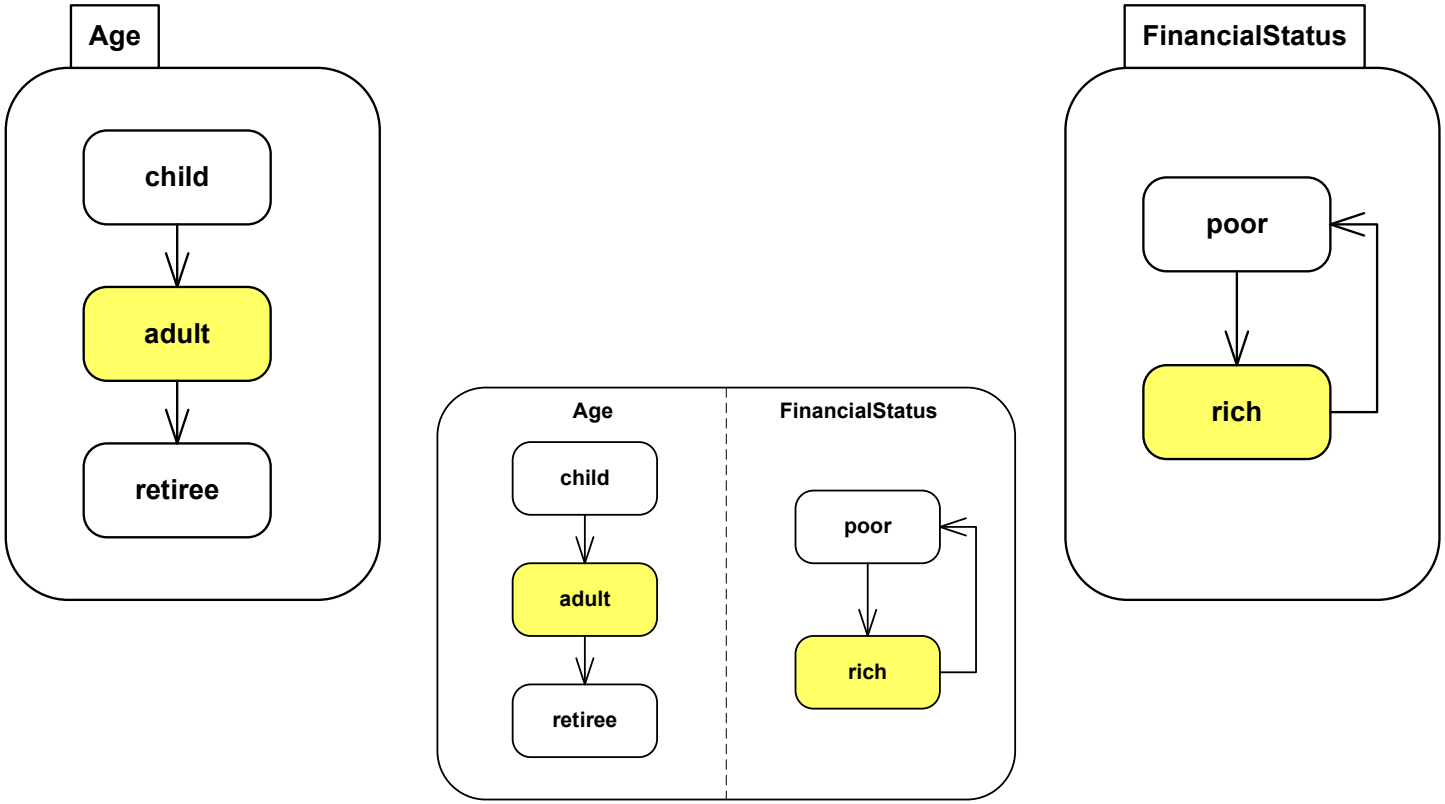
- Multiple simultaneous perspectives on the same entity





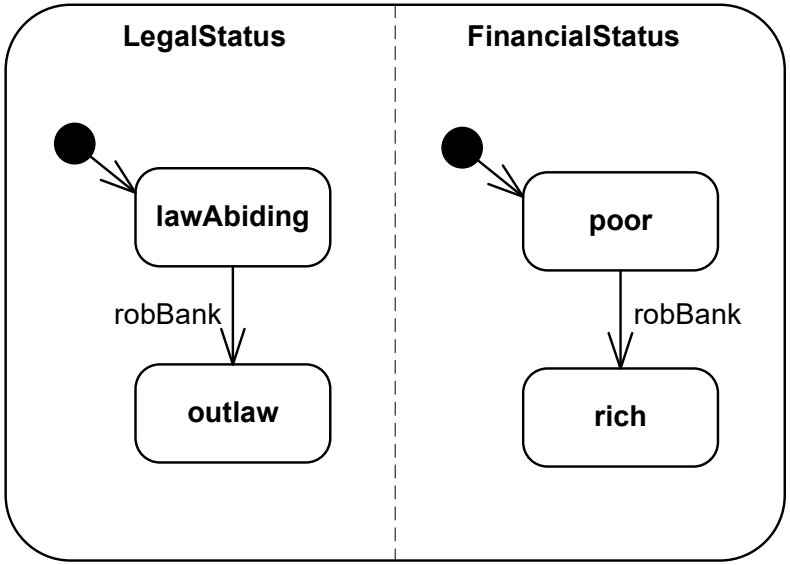
Parallelism: States with orthogonal regions

- Combine multiple simultaneous descriptions



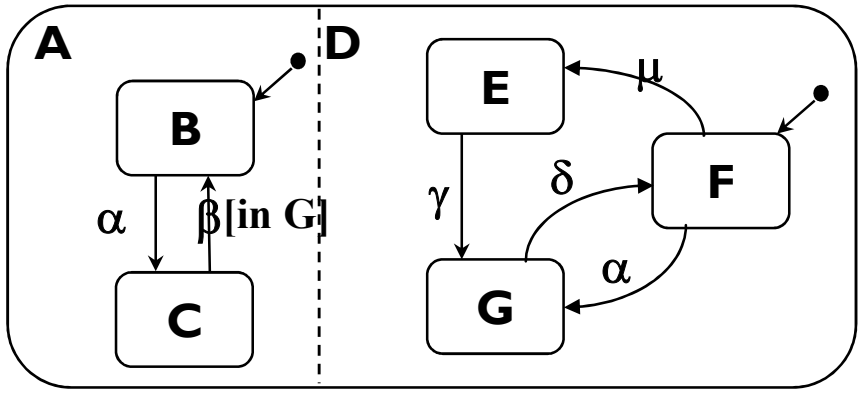
Parallelism: States with orthogonal regions

- All mutually orthogonal regions detect the same events and respond to them "simultaneously"
  - usually reduces to interleaving of some kind

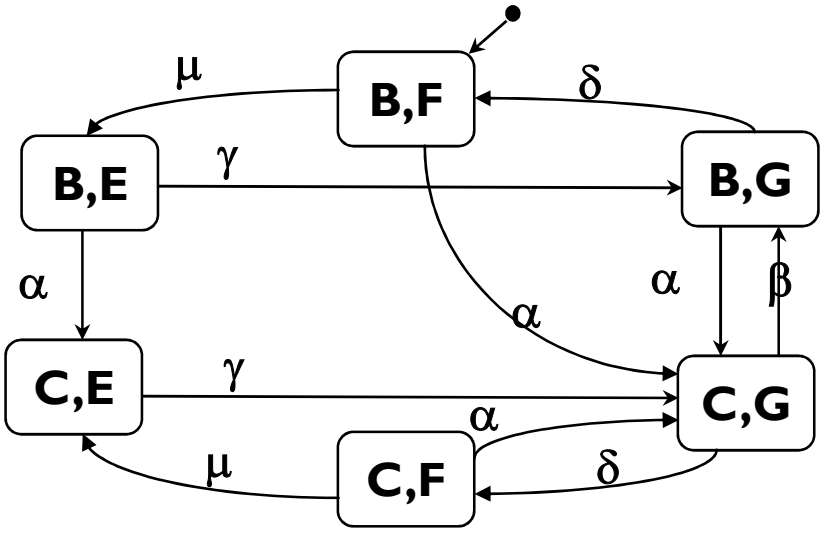


“Flat” vs. Parallel State Machines

- Every parallel machine can be transformed into a sequential machine:

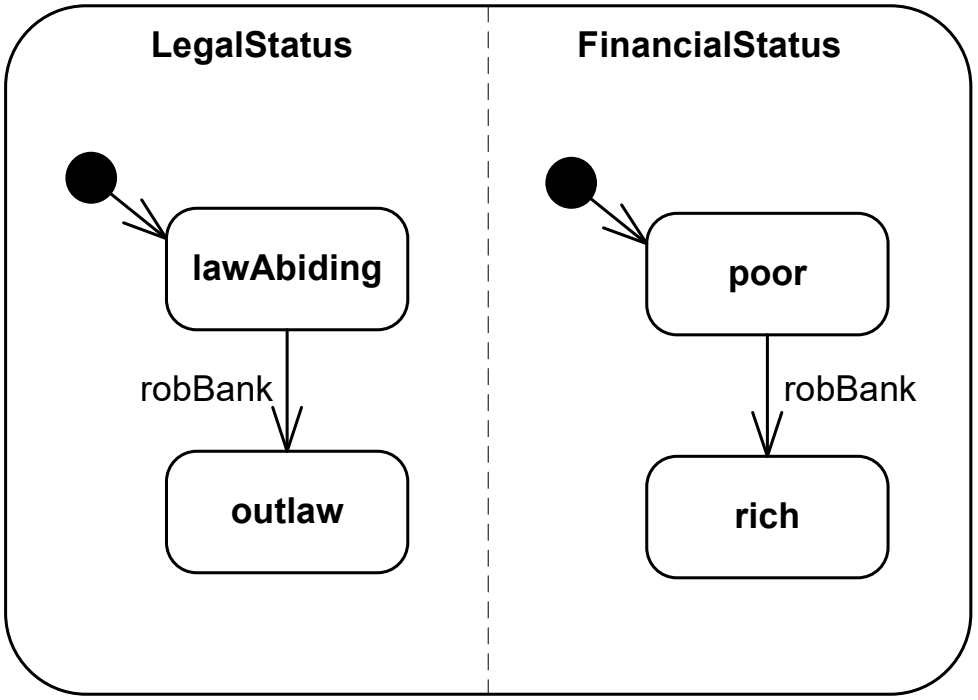


With Orthogonal Regions



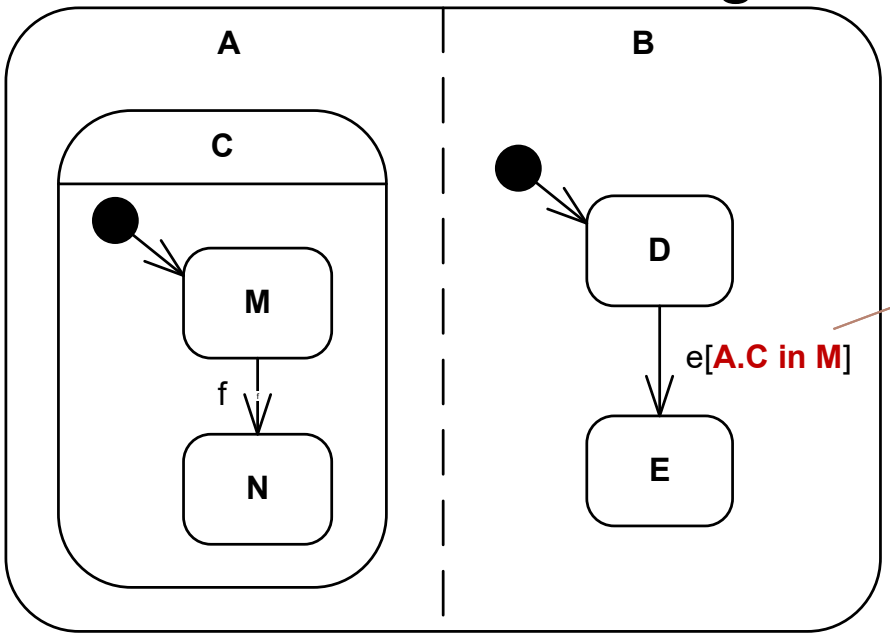
Without Orthogonal Regions

Exercise 5:
Rewrite this without parallel regions



Synchronization across orthogonal regions

- Orthogonal regions/states can be synchronized via conditions of the form "region in a state"



This transition can only be fired when A.C is in M state

- **Note:** In Yakindu the syntax for checking if region A.C is in state M is as follows: [active(A.C.M)]

Readings & Resources

- Last week: Blaha & Rumbaugh, Chapter 5
- **This week:** Blaha & Rumbaugh, Chapter 6