CS156: Topics

- Verification of **sequential** programs.
  - No concurrency.
  - Programs (should) always terminate.
  - Observable at start (input) and end (output) of execution.
- Logical foundations:
  - FOL.
  - Invariants and ranking functions.
  - Verification conditions.
  - Decision procedures.
  - Invariant generation.
  - Induction.

CS256: Topics

- Verification of **reactive systems**.
  - Highly concurrent.
    Concept of **fairness**.
    Properties: **mutual exclusion**, **freedom from deadlock**.
  - Programs need not terminate (*e.g.*, OS, web server).
    But some components must terminate (*e.g.*, IO handler).
  - Observable throughout execution.
    And the environment affects execution.
- Logical foundations: Everything from CS156 *plus*
  - **temporal logics**
    linear (LTL), branching (CTL), alternating (ATL) time
  - **automata theory** and connection with temporal logics
    infinite strings (linear) and trees (branching, alternating)

PRIME

$$\textbf{local } y: \textbf{ integer where } y = 1$$
$$\ell_0: \textbf{ loop forever do}$$
$$\left[\begin{array}{l} \vdots \\ \ell_5: \textbf{print } y \\ \ell_6: \\ \vdots \\ \ell_{10}: y \leftarrow y + 1 \\ \vdots \end{array}\right]$$

Output: 2,3,5,7,11,13, ...

- only primes: $\quad \Box[at\_\ell_5 \rightarrow \text{prime}(y)]$
- all primes: $\quad \forall u\ (\text{prime}(u)\ \rightarrow\ \Diamond[at\_\ell_5 \wedge y = u])$
- monotonicity (correct order):

$$\forall u\ [(at\_\ell_6 \wedge y = u)\ \rightarrow\ \Box(at\_\ell_5 \rightarrow y > u)]$$

BAKERY$[2]$

local $\quad y_1, y_2 \quad$: **integer where** $y_1 = 0, y_2 = 0$

$$
P_1 :: \left[\begin{array}{l}
\textbf{loop forever do} \\
\quad \left[\begin{array}{ll}
\ell_0: & \textbf{noncritical} \\
\ell_1: & y_1 := y_2 + 1 \\
\ell_2: & \textbf{await } y_2 = 0 \vee y_1 \leq y_2 \\
\ell_3: & \textbf{critical} \\
\ell_4: & y_1 := 0
\end{array}\right]
\end{array}\right]
$$

$$
||
$$

$$
P_2 :: \left[\begin{array}{l}
\textbf{loop forever do} \\
\quad \left[\begin{array}{ll}
m_0: & \textbf{noncritical} \\
m_1: & y_2 := y_1 + 1 \\
m_2: & \textbf{await } y_1 = 0 \vee y_2 < y_1 \\
m_3: & \textbf{critical} \\
m_4: & y_2 := 0
\end{array}\right]
\end{array}\right]
$$

Requirements for BAKERY[2]

- Mutual exclusion

$$\boxed{\Box\neg(\ell_3 \ \wedge \ m_3)}$$

  The two processes are not in the critical section simultaneously.

- One-bounded overtaking

$$\boxed{\ell_2 \ \Rightarrow \ \neg m_3 \,\mathcal{W}\, m_3 \,\mathcal{W}\, \neg m_3 \,\mathcal{W}\, \ell_3}$$

  Once P_1 waits to get access, P_2 can enter its critical section at most once.

- Progress

$$\boxed{\ell_1 \ \Rightarrow \Diamond \ell_3}$$

  Once P_1 shows interest in entering its critical section, it eventually gets access to the critical section.

CS256: Administration

- TTh 11:00-12:15, Gates B12
- Instructor: Zohar Manna
  TA: Matteo Slanina
- Text:

  The Temporal Verification of Reactive Systems: Safety
  Zohar Manna and Amir Pnueli

- Prerequisites: CS103, CS156, or equivalent background