Automata / state machines

- What is an automaton?
- What are behaviors of automata? Executions
- What are safety requirements?
- Why is safety analysis of automata hard?
- Basic method for proving safety of all executions

An automaton / state machine $A$ is defined by:

- A set of states $\mathcal{Q}$
- A start set $\mathcal{Q}_0 \subseteq \mathcal{Q}$
- A set of transitions $\mathcal{D} \subseteq \mathcal{Q} \times \mathcal{Q}$

**Example:** Cruise control logic

$A:$

Drawing states and arrows gets messy

We use programs to describe automata!
Determinism / Non determinism

Deterministic automaton  Non determinism

- Models arbitrary choice
  - e.g. Failure, decisions

So given a state \( q \in Q \) there are many possible "next" states.
Example 2 (Slides first) Car 1  Car 2

\[ \begin{align*}
\text{State} & \quad \text{vector} \\
0 & \quad x_{10} \quad x_{20} \quad \rightarrow x
\end{align*} \]

\[ G = \mathbb{R}^4 \]

\[ G_0 = \tilde{x}_0 := \]

\[ \mathcal{D} \subseteq \mathbb{R}^4 \times \mathbb{R}^4 \]

\[ \text{Eg. if } x_2 - x_1 < d_s \]

\[ v_1 := \max(0, v_1 - a_b) \]

\[ \text{else } v_1 := v_1 \]

\[ x_2 := x_2 + v_2 \]

\[ x_1 := x_1 + v_1 \]

What does this really define?

\[ \mathcal{F} = \{ (x, x') \in \mathbb{R}^4 \times \mathbb{R}^4 \} \]

This representation of \( \mathcal{F} \) will be important when we want to work with sets of states.
This example is deterministic

Ex: Make it non deterministic

Execution

An execution is a particular behavior of \( A \)

\[ \alpha = q_0, q_1, \ldots, q_k \quad \text{finite or infinite} \]

(i) \( q_0 \in Q \)

(ii) \( \forall i \quad (q_i, q_{i+1}) \in \Delta \)

Nondeterministic automata can have many executions e.g. \( \text{off, cl, cl, ch} \)

\( \text{off, cl, off, cl} \)

If \( \alpha \) is an execution \( \alpha = q_0, q_1, \ldots \) the \( i \text{th} \) state in \( \alpha \) is written as \( \alpha[i] = q_i \)
Requirements (Safety & Unsafety)

A requirement for A is any statement/formula involving the states in $\mathcal{S}$ that is satisfied by all executions of A.

Example: • "Car 1 should never come within 0.5m of Car 2."

$P_1 : \"x_2 - x_1 \geq 0.5\"$

$P_1 = \{x \in \mathbb{R}^4 | x \cdot x_2 - x \cdot x_1 \geq 0.5\}$

• "Car should never exceed speed limit" : $U$

• "always \geq 25 mpg" : $P_2$

With a single test/execution we can check whether every state in the execution satisfies the requirement or not, or equivalently does not satisfy the unsafety requirement ($U$).
To generate individual tests we can “just run” the program representing $\mathcal{F}$.

But, to cover all behaviors/executions we have to work with sets of states.

**Def 1** For $\mathcal{A} = \langle \mathcal{O}, \mathcal{O}_0, \delta \rangle$ any set $S \subseteq \mathcal{O}$

$$Post(S) = \{ \}$$

**Exercise (monotonic)**

**Def 2**

$$Post^0(S) := S$$

$$Post^k(S) := Post(\text{Post}^{k-1}(S)) \quad k > 0$$

**Proposition** The set of states $R^k$ that $R$ can reach after $k$-transitions (at the end of executions of length $k$) is $Post^k(\mathcal{O}_0)$.

**ICYAI** In Case You Are Interested
**Proof**: \( R^k \) = states that are at the end of executions of length \( k \).

We have to show (i) \( R^k \subseteq Post^k(G_0) \) and (ii) \( Post^k(G_0) \subseteq R^k \).

(i) Proof by induction on \( k \).

**Base**. \( R^0 = G_0 \) by def of execution of length 0.

\[ Post^0(G_0) = G_0 \] by def of \( Post^k \)

Therefore \( R^0 \subseteq Post^0(G_0) \).

**Induction**. Suppose \( R^k \subseteq Post^k(G_0) \) \(-1\)

Now consider \( R^{k+1} \) for any \( x \in R^{k+1} \) \( \exists \) execution \( f \) length \( k \) with last state \( x' \in R^k \) by \(-1\)

and \( (x', x) \in \delta \)

\[ \Rightarrow x' \in Post(x) \subseteq Post(R^k) \leq Post(Post^k(G_0)) \text{ by } \text{-1} \]

\[ \Rightarrow x' \in Post^{k+1}(G_0) \]

\[ R^{k} \subseteq Post^{k}(G_0) \text{ and } k \]

(ii) \( Post^k(G_0) \subseteq R^k \)

**Base case**: same as above.

**Induction**: hypothesis \( Post^k(G_0) \subseteq R^k \) \(-2\)
Now consider \( x \in \text{Post}^{k+1}(\mathcal{G}_0) \) there must be \( \alpha' \) an execution of length \( k \) with last state \( x' \) in \( \text{Post}^k(\mathcal{G}_0) \) by (2) and \( (x',x) \in \mathcal{D} \).

Then \( \alpha'x \) is an exec of length \( k+1 \) with \( x \in R^{k+1} \) \( \text{Post}^{k+1}(\mathcal{G}) \subseteq R^{k+1} \) \( \square \)

To prove that all executions are safe forever we would want to compute

\[ \text{Prop2} \]

If we can find a set \( I \subseteq \mathcal{G} \) such that

\[ (i) \quad \mathcal{G}_0 \subseteq I \]
\[ (ii) \quad \text{Post}(I) \subseteq I \]

Idea. Approximate \( \text{Post}^R(\mathcal{G}_0) \)
Then \( \text{Post}^k(Q_0) \subseteq I \leq \forall k \).

I over approximates \( \text{Post}^k(Q_0) \).

Proof. Induction on \( k \).

Such an \( I \) is called an Inductive invariant of \( A \).

- If we can find \( I \) and \( I \cap \text{Unsat} = \emptyset \)
  we have shown that all executions of \( A \) are safe.