Automata on Finite Words

- Deterministic and Non-deterministic Finite Automata (DFA, NFA)
- A word $\omega$ is recognized by a NFA (or a DFA) if it admits (at least) one run which ends on an acceptance state.
- The language recognized by $A$, $L(A)$, is the set of recognized words.

Theorem (Equivalence DFA and NFA)
The sets of languages recognized by DFA and NFA are the same. Moreover, it is possible to transform a NFA into a DFA which recognizes the same language.

This set of languages is called regular languages (and correspond to: $L = a \in \Sigma | L_1 \cup L_2 | L_1 + L_2 | L_1^*$).
Automata on Finite Words

- Deterministic and Non-deterministic Finite Automata (DFA, NFA)
  \[ \Rightarrow \text{A word } \omega \text{ is recognized by a NFA (or a DFA) if it admits (at least) one run which ends on an acceptance state.} \]
- The language recognized by \( A \), \( L(A) \), is the set of recognized words.

**Theorem (Equivalence DFA and NFA)**

-The sets of language recognized by DFA and NFA are the same. Moreover, it is possible to transform a NFA into a DFA which recognizes the same language.

- This set of languages is called regular languages
  (and correspond to: \( L = a \in \Sigma \mid L_1.L_2 \mid L_1 + L_2 \mid L^* \))
Automata on Infinite Words

- \( \mathcal{L} \) is \( \omega \)-regular language iff \( \mathcal{L} = \mathcal{L}_\omega(E_1.F_1^\omega + \cdots + E_n.F_n^\omega) \)
- A \( \omega \)-regular property \( P \) over \( AP \) is a LT property over \( AP \), such that \( P \) is a \( \omega \)-regular language over \( 2^{AP} \).
Automata on Infinite Words

- \( \mathcal{L} \) is \( \omega \)-regular language iff \( \mathcal{L} = \mathcal{L}_\omega(\mathcal{E}_1.F_1^\omega + \cdots + \mathcal{E}_n.F_n^\omega) \)

- A \( \omega \)-regular property \( P \) over \( AP \) is a LT property over \( AP \), such that \( P \) is a \( \omega \)-regular language over \( 2^{AP} \).

- Non-deterministic Büchi Automata (NBA)
  - A word \( \sigma \) is recognized by a NBA if it admits (at least) one run which goes through acceptance states infinitely often.

Variants: Deterministic/Generalized Büchi Automata (DBA/GNBA).

Theorem (Equivalences)
The languages recognized by NBA and GNBA are \( \omega \)-regular languages.

DBA are less expressive than NBA. (cf: \( (A + B)^*B^\omega \)).
Automata on Infinite Words

- \( \mathcal{L} \) is \( \omega \)-regular language iff \( \mathcal{L} = \mathcal{L}_\omega(E_1.F_1^\omega + \cdots + E_n.F_n^\omega) \)
- A \( \omega \)-regular property \( P \) over \( AP \) is a LT property over \( AP \), such that \( P \) is a \( \omega \)-regular language over \( 2^{AP} \).

- Non-deterministic Büchi Automata (NBA)
  - A word \( \sigma \) is recognized by a NBA if it admits (at least) one run which goes through acceptance states infinitely often.

- Variants: Deterministic/Generalized Büchi Automata (DBA/GNBA).

Theorem (Equivalences)

- The languages recognized by NBA and GNBA are \( \omega \)-regular languages.
- DBA are less expressive than NBA. (cf: \((A + B)^*B^\omega\)).
Model checking 1

**Regular safety properties:**

- Does $TS$ verifies the regular property $P$?
- We build $\mathcal{A}$ NFA recognizing the bad prefixes of $P$.

Then, we build $(TS \otimes \mathcal{A})$:

$TS \models P \iff \text{trace}_{\text{fin}}(TS) \cap L(\mathcal{A}) = \emptyset \iff TS \otimes \mathcal{A} \models P_{inv}(\mathcal{A})$

(with $P_{inv}(\mathcal{A}) = \neg F$ an invariant property).
Regular safety properties:

- Does \( TS \) verifies the regular property \( P \)?
- We build \( A \) NFA recognizing the bad prefixes of \( P \).
  
  Then, we build \((TS \otimes A)\):
  
  \( TS \models P \iff \text{trace}_{\text{fin}}(TS) \cap L(A) = \emptyset \iff TS \otimes A \models P_{\text{inv}}(A) \)  
  (with \( P_{\text{inv}}(A) = \neg F \) an invariant property).

How to check that \( TS \models P_{\text{inv}}(\Phi) \)?

\( \iff \) Does it exist a reachable \( s \in TS \) which violates \( \Phi \)?

- We can check this by using a DFS algorithm on the graph \( TS \).
• \( \omega \)-regular safety properties:
  • Does \( TS \) verifies the \( \omega \)-regular property \( P \)?
  • We build \( \mathcal{A} \) NBA with \( \mathcal{L}(\mathcal{A}) = \bar{P} \)
  → Then, we build \( (TS \otimes \mathcal{A}) \):
    \( TS \models P \iff \text{trace}(TS) \cap \mathcal{L}_\omega(\mathcal{A}) = \emptyset \iff TS \otimes \mathcal{A} \models P_{\text{pers}}(\mathcal{A}) \)
    (with \( P_{\text{pers}}(\mathcal{A}) = "\text{eventually forever } \neg F" \), a persistence property).
\(\omega\)-regular safety properties:

- Does \(TS\) verifies the \(\omega\)-regular property \(P\)?
- We build \(A\) NBA with \(L(A) = \bar{P}\)

\[TS \models P \iff \text{trace}(TS) \cap L_\omega(A) = \emptyset \iff TS \otimes A \models P_{\text{pers}}(A)\]
(with \(P_{\text{pers}}(A) = "\text{eventually forever } \neg F"\), a persistence property).

**How to check that** \(TS \models P_{\text{pers}}(\Phi)\)?

\[\iff\text{ Does it exist a reachable } s \in TS:\]

- which violates \(\Phi\),
- and which is inside a cycle of \(TS\)?

- We check this by using a combination of DFS algorithm on \(TS\).