
Closure Properties & Emptiness Test for DFA

Martin Fränzle

Informatics and Mathematical Modelling
The Technical University of Denmark

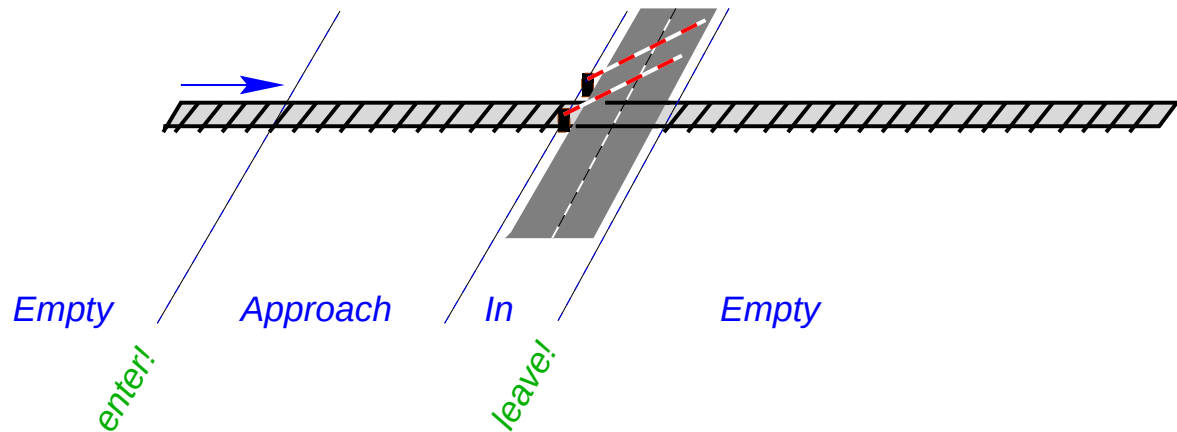
02140 Languages and Parsing, MF, Fall 2004 – p.1/14

**Deciding non-trivial properties
using automata:**

An example

02140 Languages and Parsing, MF, Fall 2004 – p.2/14

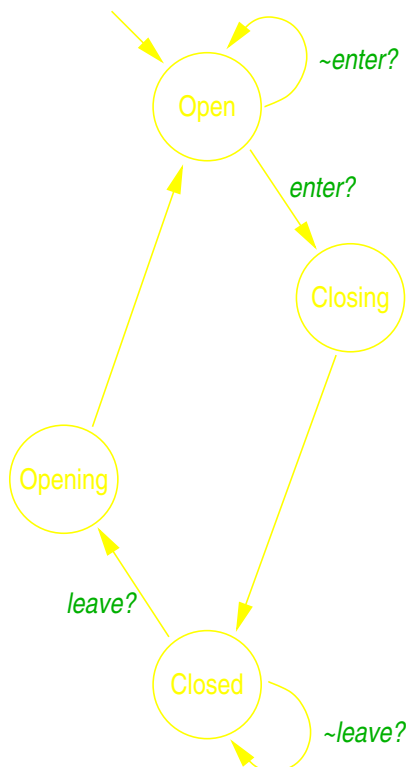
Explicit state model-checking



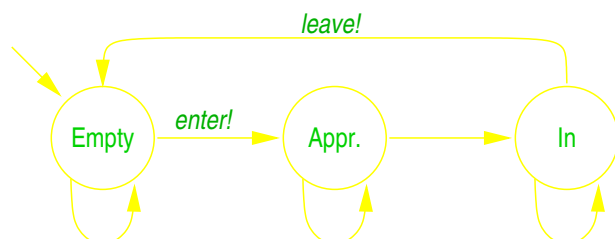
Safety requirement: Gate has to be closed whenever a train is in “In”.

02140 Languages and Parsing, MF, Fall 2004 – p.3/14

The gate model

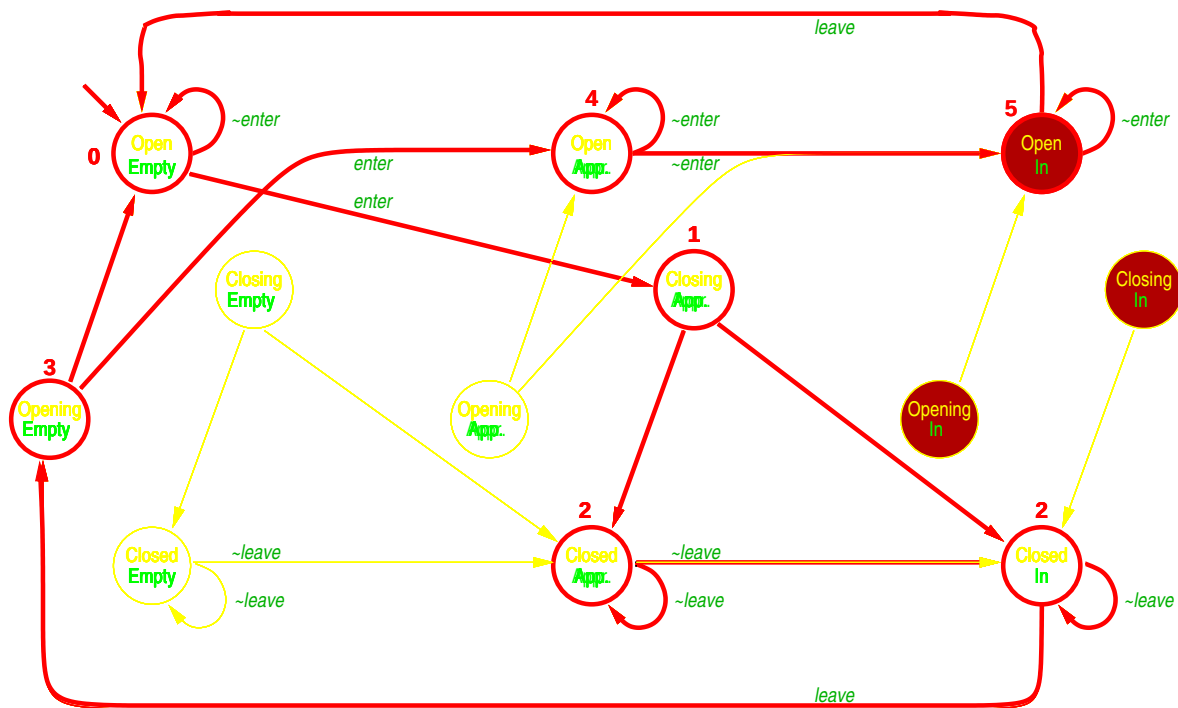


Track model
— safe abstraction —



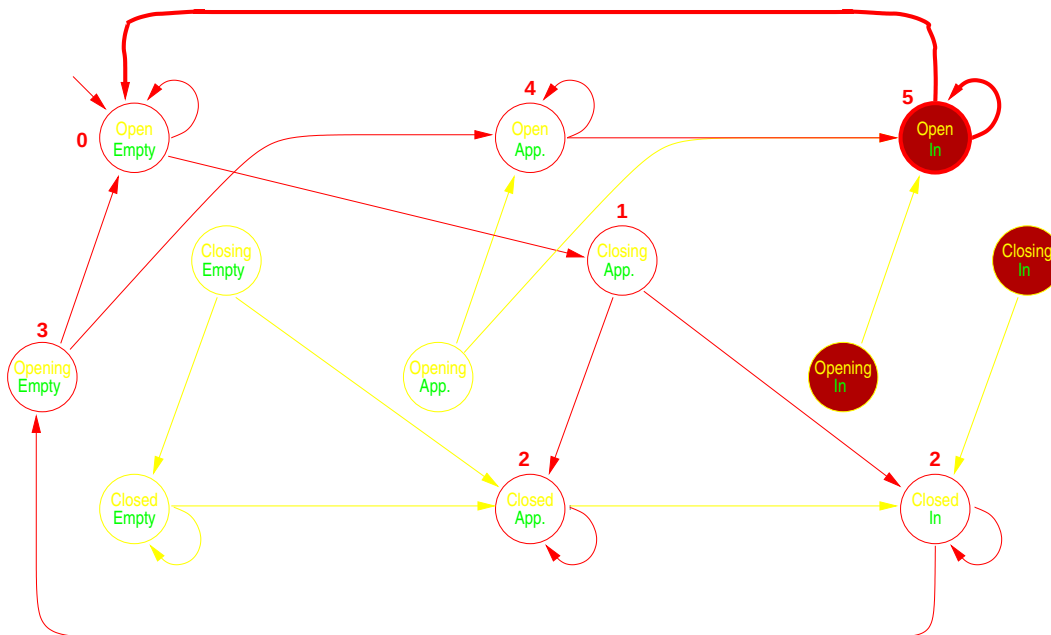
02140 Languages and Parsing, MF, Fall 2004 – p.4/14

Automatic check



02140 Languages and Parsing, MF, Fall 2004 – p.5/14

Verification result



Stimuli: Empty, Approach, In, Empty, Approach, In.
Gate reaction: Open, Closing, Closed, Opening, Open, Open.

02140 Languages and Parsing, MF, Fall 2004 – p.6/14

Decision procedures:

Language emptiness

02140 Languages and Parsing, MF, Fall 2004 – p.7/14

The coloring algorithm

The set R of reachable states of a (D/N/ ϵ N)FA $(Q, \Sigma, \delta, q_0, F)$ is the *smallest* set satisfying

Base case: $q_0 \in R$,

Recursion: if $p \in R$ and there is $a \in \Sigma$ ($a \in \Sigma \cup \{\epsilon\}$, respectively) s.t.
 $q = \delta(p, a)$ (or $q \in \delta(p, a)$ respectively) then $q \in R$.

Adding some bookkeeping to avoid reexamination of already visited states, above rules yield an algorithm for computing the reachable states.

Complexity: $O(m)$, where m is the number of state pairs connected via transitions in the automaton; $m \leq n^2$ when n is the number of states.

02140 Languages and Parsing, MF, Fall 2004 – p.8/14

Language emptiness test

Corollary: It is decidable (i.e., there is an algorithm that decides) whether the language of a (D/N/ ϵ N)FA is empty.

Algorithm: Given a (D/N/ ϵ N)FA $A = (Q, \Sigma, \delta, q_0, F)$ do

1. compute the set R of reachable states of A ,
2. check whether $R \cap F = \emptyset$.

The language of A is empty iff $R \cap F = \emptyset$, i.e. iff no final state is reachable.

Finite automata

Intersection and complementation

Intersection: Product construction

Thm: Given two DFA $A = (Q, \Sigma, \delta_A, q_0, F)$ and $B = (R, \Sigma, \delta_B, r_0, G)$, the DFA $A \times B = (S, \Sigma, \delta_{AB}, s_0, H)$ defined by

- $S \stackrel{\text{def}}{=} Q \times R$, States are pairs, $|S| = |Q| \cdot |R|$
- $\delta_{AB}((q, r), a) \stackrel{\text{def}}{=} (\delta_A(q, a), \delta_B(r, a))$ for each $s = (q, r) \in S$ and each $a \in \Sigma$,
- $s_0 \stackrel{\text{def}}{=} (q_0, r_0)$,
- $H \stackrel{\text{def}}{=} \{(q, r) \mid q \in F, r \in G\}$.

accepts the language $L(A \times B) = L(A) \cap L(B)$.

Def: $A \times B$ is called the “automaton product” of A and B .

02140 Languages and Parsing, MF, Fall 2004 – p.11/14

Complementation

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

Thm: The DFA $A' = (Q, \Sigma, \delta, q_0, Q \setminus F)$ accepts the language $\Sigma^* \setminus L(A)$.

I.e., complementing the set of accepting states complements the language recognized.

Prf:

$$\begin{aligned} L(A') &= \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in Q \setminus F\} \\ &= \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \notin F\} \\ &= \{w \in \Sigma^* \mid w \notin L(A)\} \\ &= \Sigma^* \setminus L(A) \end{aligned}$$

02140 Languages and Parsing, MF, Fall 2004 – p.12/14

Further decision procedures:

Combining closure properties and emptiness test

02140 Languages and Parsing, MF, Fall 2004 – p.13/14

Boolean combinations of languages

Using intersection and complement, arbitrary Boolean combinations of languages can be built:

- **Language union:** $L(A) \cup L(B) = \overline{\overline{L(A)} \cap \overline{L(B)}} = ((A' \times B')')$
- **Lang. difference:** $L(A) \setminus L(B) = L(A) \cap \overline{L(B)} = L(A \times B')$
- ...

Combining these with the emptiness test, further properties can be decided:

- **Lang. inclusion:** To test for $L(A) \subseteq L(B)$, test $L(A) \setminus L(B) = L(A \times B')$ for emptiness.
- **Lang. equivalence:** Tests for $L(A) = L(B)$ can be reduced to two language inclusion tests.

02140 Languages and Parsing, MF, Fall 2004 – p.14/14