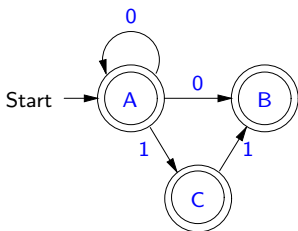


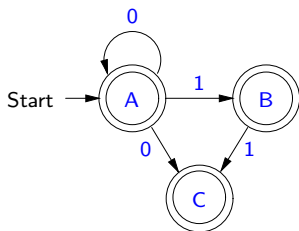
3.5: Isomorphism of Finite Automata

Let M and N be the finite automata



(M)

and

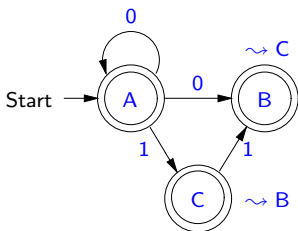


(N)

How are M and N related?

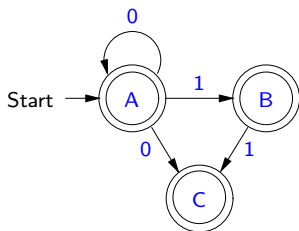
3.5: Isomorphism of Finite Automata

Let M and N be the finite automata



(M)

and



(N)

How are M and N related? Although they are not equal, they do have the same “structure”, in that M can be turned into N by replacing A , B and C by A , C and B , respectively. When FAs have the same structure, we will say they are “isomorphic”.

Definition of Isomorphism

An *isomorphism* h from an FA M to an FA N is a bijection from Q_M to Q_N such that

- $h s_M =$
- $\{ h q \mid q \in A_M \} =$
- $\{ (h q), x \rightarrow (h r) \mid q, x \rightarrow r \in T_M \} =$

Definition of Isomorphism

An *isomorphism* h from an FA M to an FA N is a bijection from Q_M to Q_N such that

- $h s_M = s_N$;
- $\{ h q \mid q \in A_M \} =$
- $\{ (h q), x \rightarrow (h r) \mid q, x \rightarrow r \in T_M \} =$

Definition of Isomorphism

An *isomorphism* h from an FA M to an FA N is a bijection from Q_M to Q_N such that

- $h s_M = s_N$;
- $\{ h q \mid q \in A_M \} = A_N$; and
- $\{ (h q), x \rightarrow (h r) \mid q, x \rightarrow r \in T_M \} =$

Definition of Isomorphism

An *isomorphism* h from an FA M to an FA N is a bijection from Q_M to Q_N such that

- $h s_M = s_N$;
- $\{ h q \mid q \in A_M \} = A_N$; and
- $\{ (h q), x \rightarrow (h r) \mid q, x \rightarrow r \in T_M \} = T_N$.

Definition of Isomorphism

An *isomorphism* h from an FA M to an FA N is a bijection from Q_M to Q_N such that

- $h s_M = s_N$;
- $\{ h q \mid q \in A_M \} = A_N$; and
- $\{ (h q), x \rightarrow (h r) \mid q, x \rightarrow r \in T_M \} = T_N$.

We define a relation **iso** on **FA** by: $M \text{ iso } N$ iff there is an isomorphism from M to N . We say that M and N are *isomorphic* iff $M \text{ iso } N$.

Definition of Isomorphism

An *isomorphism* h from an FA M to an FA N is a bijection from Q_M to Q_N such that

- $h s_M = s_N$;
- $\{h q \mid q \in A_M\} = A_N$; and
- $\{(h q), x \rightarrow (h r) \mid q, x \rightarrow r \in T_M\} = T_N$.

We define a relation **iso** on **FA** by: $M \text{ iso } N$ iff there is an isomorphism from M to N . We say that M and N are *isomorphic* iff $M \text{ iso } N$.

Consider our example FAs M and N , and let h be the function

$$\{(A, A), (B, C), (C, B)\}.$$

Then h is an isomorphism from M to N . Hence $M \text{ iso } N$.

Properties of Isomorphism

Clearly, if M and N are isomorphic, then they have the same alphabet.

Proposition 3.5.1

*The relation **iso** is reflexive on **FA**, symmetric and transitive.*

Properties of Isomorphism

Proposition 3.5.2

Suppose M and N are isomorphic FAs. Then $L(M) \subseteq L(N)$.

Proof.

□

Properties of Isomorphism

Proposition 3.5.2

Suppose M and N are isomorphic FAs. Then $L(M) \subseteq L(N)$.

Proof. Let h be an isomorphism from M to N . Suppose $w \in L(M)$. Then, there is a labeled path

$$lp = q_1 \xrightarrow{x_1} q_2 \xrightarrow{x_2} \cdots q_n \xrightarrow{x_n} q_{n+1},$$

such that $w = x_1 x_2 \cdots x_n$, lp is valid for M , $q_1 = s_M$ and $q_{n+1} \in A_M$. Let

$$lp' = h q_1 \xrightarrow{x_1} h q_2 \xrightarrow{x_2} \cdots h q_n \xrightarrow{x_n} h q_{n+1}.$$

Then the label of lp' is w , lp' is valid for N , $h q_1 = h s_M = s_N$ and $h q_{n+1} \in A_N$, showing that $w \in L(N)$. \square

Properties of Isomorphism

Proposition 3.5.3

Suppose M and N are isomorphic FAs. Then $M \approx N$.

Proof.

□

Properties of Isomorphism

Proposition 3.5.3

Suppose M and N are isomorphic FAs. Then $M \approx N$.

Proof. Since $M \text{ iso } N$, we have that $N \text{ iso } M$, by Proposition 3.5.1.

□

Properties of Isomorphism

Proposition 3.5.3

Suppose M and N are isomorphic FAs. Then $M \approx N$.

Proof. Since $M \text{ iso } N$, we have that $N \text{ iso } M$, by Proposition 3.5.1. Thus, by Proposition 3.5.2, we have that $L(M) \subseteq L(N) \subseteq L(M)$. Hence $L(M) = L(N)$, i.e., $M \approx N$. \square

Renaming States

The function **renameStates** takes in a pair (M, f) , where $M \in \mathbf{FA}$ and f is a bijection from Q_M to some set of symbols, and returns the **FA** produced from M by renaming M 's states using the bijection f .

Proposition 3.5.4

*Suppose M is an FA and f is a bijection from Q_M to some set of symbols. Then **renameStates** (M, f) iso M .*

Renaming States

The function **renameStates** takes in a pair (M, f) , where $M \in \mathbf{FA}$ and f is a bijection from Q_M to some set of symbols, and returns the **FA** produced from M by renaming M 's states using the bijection f .

Proposition 3.5.4

Suppose M is an FA and f is a bijection from Q_M to some set of symbols. Then **renameStates** (M, f) iso M .

The following function is a special case of **renameStates**. The function **renameStatesCanonically** $\in \mathbf{FA} \rightarrow \mathbf{FA}$ renames the states of an FA M to:

- $A, B, \text{etc.}$, when the automaton has no more than 26 states (the smallest state of M will be renamed to A , the next smallest one to B , etc.); or
- $\langle 1 \rangle, \langle 2 \rangle, \text{etc.}$, otherwise.

An Algorithm for Finding Isomorphisms

The book presents and proves the correctness of a relatively simple algorithm for finding an isomorphism from one FA to another, if one exists, and for indicating that there are no such isomorphisms, otherwise.

Isomorphism Finding/Checking in Forlan

The Forlan module **FA** also defines the functions

```
val isomorphism           : fa * fa * sym_rel -> bool
val findIsomorphism      : fa * fa -> sym_rel
val isomorphic           : fa * fa -> bool
val renameStates         : fa * sym_rel -> fa
val renameStatesCanonically : fa -> fa
```

Forlan Examples

Suppose that `fa1` and `fa2` have been bound to our example finite automata M and N , respectively. Then, here are some example uses of the above functions:

```
- val rel = FA.findIsomorphism(fa1, fa2);  
val rel = - : sym_rel  
- SymRel.output("", rel);  
(A, A), (B, C), (C, B)  
val it = () : unit  
- FA.isomorphism(fa1, fa2, rel);  
val it = true : bool  
- FA.isomorphic(fa1, fa2);  
val it = true : bool
```

Forlan Examples

```
- val rel' = FA.findIsomorphism(fa1, fa1);  
val rel' = - : sym_rel  
- SymRel.output("", rel');  
(A, A), (B, B), (C, C)  
val it = () : unit  
- FA.isomorphism(fa1, fa1, rel');  
val it = true : bool  
- FA.isomorphism(fa1, fa2, rel');  
val it = false : bool
```

Forlan Examples

```
- val rel'' = SymRel.input "";  
@ (A, 2), (B, 1), (C, 0)  
@ .  
val rel'' = - : sym_rel  
- val fa3 = FA.renameStates(fa1, rel'');  
val fa3 = - : fa  
- FA.output("", fa3);  
{states} 0, 1, 2 {start state} 2  
{accepting states} 0, 1, 2  
{transitions} 0, 1 -> 1; 2, 0 -> 1 | 2; 2, 1 -> 0  
val it = () : unit
```

Forlan Examples

```
- val fa4 = FA.renameStatesCanonically fa3;
val fa4 = - : fa
- FA.output("", fa4);
{states} A, B, C {start state} C
{accepting states} A, B, C
{transitions} A, 1 -> B; C, 0 -> B | C; C, 1 -> A
val it = () : unit
- FA.equal(fa4, fa1);
val it = false : bool
- FA.isomorphic(fa4, fa1);
val it = true : bool
```