Decidability of Presburger Arithmetic via Finite Automata

Takao Yuyama

Automata and Logic Workshop in AKITA
March 26, 2019

Tokyo Institute of Technology

Table of Contents

Table of Contents

An Ideal Machine

Suppose there were a machine like this:

Are there infinitely many prime numbers?

Yes.

Is there a nontrivial integer solution for $x^2 + y^2 = 3z^2$?

No.

Is Goldbach conjecture true?

...

The machine eventually gives the correct answer to a question whether a given mathematical statement is true or not.

An Ideal Machine

Suppose there were a machine like this:

Are there infinitely many prime numbers?

Yes.

Is there a nontrivial integer solution for $x^2 + y^2 = 3z^2$?

No.

Is Goldbach conjecture true?

...

The machine eventually gives the correct answer to a question whether a given mathematical statement is true or not.

An Ideal Machine

Suppose there were a machine like this:

Are there infinitely many prime numbers?

Yes.

Is there a nontrivial integer solution for $x^2 + y^2 = 3z^2$?

No.

Is Goldbach conjecture true?

...

The machine eventually gives the correct answer to a question whether a given mathematical statement is true or not.

An Ideal Machine

Suppose there were a machine like this:

Are there infinitely many prime numbers?

Yes.

Is there a nontrivial integer solution for $x^2 + y^2 = 3z^2$?

No.

Is Goldbach conjecture true?

...

The machine eventually gives the correct answer to a question whether a given mathematical statement is true or not.

An Ideal Machine

Suppose there were a machine like this:

Are there infinitely many prime numbers?

Yes.

Is there a nontrivial integer solution for $x^2 + y^2 = 3z^2$?

No.

Is Goldbach conjecture true?

...

The machine eventually gives the correct answer to a question whether a given mathematical statement is true or not.

An Ideal Machine

Suppose there were a machine like this:

Are there infinitely many prime numbers?

Yes.

Is there a nontrivial integer solution for $x^2 + y^2 = 3z^2$?

No.

Is Goldbach conjecture true?

...

The machine eventually gives the correct answer to a question whether a given mathematical statement is true or not.

An Ideal Machine

Suppose there were a machine like this:

Are there infinitely many prime numbers?

Yes.

Is there a nontrivial integer solution for $x^2 + y^2 = 3z^2$?

No.

Is Goldbach conjecture true?

...

The machine eventually gives the correct answer to a question whether a given mathematical statement is true or not.

Q. Can we create such a machine in the future?

A. No!

Moreover, there is no such a machine even for the statements about natural numbers.

Theorem (Gödel, 1931)

The first-order theory of the natural numbers is undecidable.

In other words, there are no algorithms for determining whether a given arithmetical sentence is true or not.

Q. Can we create such a machine in the future?

A. No!

Moreover, there is no such a machine even for the statements about natural numbers.

Theorem (Gödel, 1931)

The first-order theory of the natural numbers is undecidable.

In other words, there are no algorithms for determining whether a given arithmetical sentence is true or not.

The Incompleteness Theorem

Q. Can we create such a machine in the future?

A. No!

Moreover, there is no such a machine even for the statements about natural numbers.

Theorem (Gödel, 1931)

The first-order theory of the natural numbers is undecidable.

In other words, there are no algorithms for determining whether a given arithmetical sentence is true or not.

The Incompleteness Theorem

Q. Can we create such a machine in the future?

A. No!

Moreover, there is no such a machine even for the statements about natural numbers.

Theorem (Gödel, 1931)

The first-order theory of the natural numbers is undecidable.

In other words, there are no algorithms for determining whether a given arithmetical sentence is true or not.

Table of Contents

The Aim of This Section

To clarify the meaning of the words

"first-order theory of the natural numbers"

and

"arithmetical sentence",

we introduce precise definitions from mathematical logic.

Available Symbols

Definition (vocabularies for arithmetic)

Define

$$\mathcal{L}_{\text{arith}} = \{\underline{0}, \underline{1}, +, \times, <\},$$
$$\mathcal{L}_{\text{Pres}} = \{\underline{0}, \underline{1}, +, <\}$$

where

- $\underline{0}$ and $\underline{1}$ are constant symbols,
- $+$ and $\times$ are binary function symbols,
- $<$ is a binary relation symbol.

Remark

Of course you may use $\mathcal{L}'_{\text{Pres}} = \{\underline{0}, \text{succ}, +, <\}$ instead of $\mathcal{L}_{\text{Pres}}$. However, $\mathcal{L}_{\text{Pres}}$ is more suitable for the algorithm we construct later.

Available Symbols

Definition (vocabularies for arithmetic)

Define

$$\mathcal{L}_{\text{arith}} = \{\underline{0}, \underline{1}, +, \times, <\},$$
$$\mathcal{L}_{\text{Pres}} = \{\underline{0}, \underline{1}, +, <\}$$

where

- $\underline{0}$ and $\underline{1}$ are constant symbols,
- $+$ and $\times$ are binary function symbols,
- $<$ is a binary relation symbol.

Remark

Of course you may use $\mathcal{L}'_{\text{Pres}} = \{\underline{0}, \mathtt{succ}, +, <\}$ instead of $\mathcal{L}_{\text{Pres}}$. However, $\mathcal{L}_{\text{Pres}}$ is more suitable for the algorithm we construct later.

Terms

Terms express a natural number.

Let $\mathcal{L}$ be either $\mathcal{L}_{\text{arith}}$ or $\mathcal{L}_{\text{Pres}}$.

Definition ($\mathcal{L}$-terms)

1. All variable symbols $x, y, z, \ldots$ and constant symbols $\underline{0}$, $\underline{1}$ are $\mathcal{L}$-terms.
2. If t_1 and t_2 are two $\mathcal{L}$-terms, then
   - $(t_1 + t_2)$ and $(t_1 \times t_2)$ are $\mathcal{L}_{\text{arith}}$-terms,
   - $(t_1 + t_2)$ is an $\mathcal{L}_{\text{Pres}}$-term.

Example

$\underline{0},\ \underline{1},\ x,\ y,\ ((x + \underline{1}) + y)$	:	$\mathcal{L}_{\text{arith}}$-terms and $\mathcal{L}_{\text{Pres}}$-terms
$((x \times y) \times z),\ (\underline{0} \times (x + y))$	:	$\mathcal{L}_{\text{arith}}$-terms but not $\mathcal{L}_{\text{Pres}}$-terms
$2x,\ y^3,\ \underline{1}/\underline{0},\ +)\times+xy(($	:	neither $\mathcal{L}_{\text{arith}}$-terms nor $\mathcal{L}_{\text{Pres}}$-terms

Terms

Terms express a natural number.

Let $\mathcal{L}$ be either $\mathcal{L}_{\text{arith}}$ or $\mathcal{L}_{\text{Pres}}$.

Definition ($\mathcal{L}$-terms)

1. All variable symbols $x, y, z, \ldots$ and constant symbols $\underline{0}$, $\underline{1}$ are $\mathcal{L}$-terms.
2. If t_1 and t_2 are two $\mathcal{L}$-terms, then
   - $(t_1 + t_2)$ and $(t_1 \times t_2)$ are $\mathcal{L}_{\text{arith}}$-terms,
   - $(t_1 + t_2)$ is an $\mathcal{L}_{\text{Pres}}$-term.

Example

$\underline{0},\ \underline{1},\ x,\ y,\ ((x + \underline{1}) + y)$	:	$\mathcal{L}_{\text{arith}}$-terms and $\mathcal{L}_{\text{Pres}}$-terms
$((x \times y) \times z),\ (\underline{0} \times (x + y))$	:	$\mathcal{L}_{\text{arith}}$-terms but not $\mathcal{L}_{\text{Pres}}$-terms
$2x,\ y^3,\ \underline{1}/\underline{0},\ +)\times+xy(($	:	neither $\mathcal{L}_{\text{arith}}$-terms nor $\mathcal{L}_{\text{Pres}}$-terms

Formulas (1/2)

Formulas express a condition for a free variable or a relation among free variables.

Sentences are formal correspondents to the notion of "mathematical statements."

Definition ($\mathcal{L}$-formulas)

1. If t_1 and t_2 are two $\mathcal{L}$-terms, then $t_1 = t_2$ and $t_1 < t_2$ are $\mathcal{L}$-formulas.
2. If φ and ψ are two $\mathcal{L}$-formulas, then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$ $(\varphi \to \psi)$ and $\neg\varphi$ are $\mathcal{L}$-formulas.
3. If φ is an $\mathcal{L}$-formula and x is a variable, then $\exists x\varphi$ and $\forall x\varphi$ are $\mathcal{L}$-formulas.

An $\mathcal{L}$-formula is an $\mathcal{L}$-sentence ($\approx$ "arithmetical sentence") if it has no free variables, i.e., all of the occurrences of a variable in the formula are bound by $\exists$ or $\forall$.

Formulas (2/2)

Example

All of the following are $\mathcal{L}_{\mathsf{Pres}}$-sentences and thus $\mathcal{L}_{\mathsf{arith}}$-sentences.

$$\exists x \exists y((x + y) = \underline{1} \wedge (((x + x) + x) + (y + y)) = (\underline{1} + \underline{1})),$$

$$\forall x \exists y(y < x).$$

Example

Let $\mathrm{Prime}(p)$ be an abbreviation for the $\mathcal{L}_{\mathsf{arith}}$-formula

$$(\underline{1} < p \wedge \forall u \forall v((u < p \wedge v < p) \rightarrow (u \times v) < p)).$$

Then the following are $\mathcal{L}_{\mathsf{arith}}$-sentences.

$$\forall n \exists p(n < p \wedge \mathrm{Prime}(p)),$$

$$\forall n \exists p(n < p \wedge (\mathrm{Prime}(p) \wedge \mathrm{Prime}(p + (\underline{1} + \underline{1})))),$$

$$\forall n(\exists m(((\underline{1} + \underline{1}) \times m) + (\underline{1} + \underline{1})) = n$$
$$\rightarrow \exists p \exists q((\mathrm{Prime}(p) \wedge \mathrm{Prime}(q)) \wedge n = (p + q)).$$

Formulas (2/2)

Example

All of the following are $\mathcal{L}_{\text{Pres}}$-sentences and thus $\mathcal{L}_{\text{arith}}$-sentences.

$$\exists x \exists y((x + y) = \underline{1} \wedge (((x + x) + x) + (y + y)) = (\underline{1} + \underline{1})),$$
$$\forall x \exists y(y < x).$$

Example

Let $\text{Prime}(p)$ be an abbreviation for the $\mathcal{L}_{\text{arith}}$-formula

$$(\underline{1} < p \wedge \forall u \forall v((u < p \wedge v < p) \rightarrow (u \times v) < p)).$$

Then the following are $\mathcal{L}_{\text{arith}}$-sentences.

$$\forall n \exists p(n < p \wedge \text{Prime}(p)),$$
$$\forall n \exists p(n < p \wedge (\text{Prime}(p) \wedge \text{Prime}(p + (\underline{1} + \underline{1})))),$$
$$\forall n(\exists m(((\underline{1} + \underline{1}) \times m) + (\underline{1} + \underline{1})) = n$$
$$\rightarrow \exists p \exists q((\text{Prime}(p) \wedge \text{Prime}(q)) \wedge n = (p + q)).$$

Satisfaction (1/2)

Here we fix a meaning of our symbols in an obvious way.

Definition (interpretation)

For any $\mathcal{L}_{\text{arith}}$-term t without free variables (i.e., built from $\underline{0}$, $\underline{1}$, $+$ and $\times$), define a natural number $t^{\mathbb{N}}$ by the following rules.

1. $\underline{0}^{\mathbb{N}} = 0$,
2. $\underline{1}^{\mathbb{N}} = 1$,
3. $(t_1 + t_2)^{\mathbb{N}} = t_1^{\mathbb{N}} + t_2^{\mathbb{N}}$,
4. $(t_1 \times t_2)^{\mathbb{N}} = t_1^{\mathbb{N}} \cdot t_2^{\mathbb{N}}$.

Satisfaction (2/2)

Definition (satisfaction relation)

For any $\mathcal{L}_{\text{arith}}$-sentence φ, define $\mathbb{N} \models \varphi$ by the following rules.

1. $\mathbb{N} \models t_1 = t_2 \iff t_1^{\mathbb{N}} = t_2^{\mathbb{N}}$,
2. $\mathbb{N} \models t_1 < t_2 \iff t_1^{\mathbb{N}} < t_2^{\mathbb{N}}$,
3. $\mathbb{N} \models (\varphi \wedge \psi) \iff \mathbb{N} \models \varphi$ and $\mathbb{N} \models \psi$,
4. $\mathbb{N} \models (\varphi \vee \psi) \iff \mathbb{N} \models \varphi$ or $\mathbb{N} \models \psi$,
5. $\mathbb{N} \models (\varphi \to \psi) \iff$ not $\mathbb{N} \models \varphi$ or $\mathbb{N} \models \psi$,
6. $\mathbb{N} \models \neg\varphi \iff$ not $\mathbb{N} \models \varphi$,
7. $\mathbb{N} \models \exists x \varphi(x) \iff$ there exists $n \in \mathbb{N}$ such that $\mathbb{N} \models \varphi(\underbrace{\underline{1} + \cdots + \underline{1}}_{n})$,
8. $\mathbb{N} \models \forall x \varphi(x) \iff \mathbb{N} \models \varphi(\underbrace{\underline{1} + \cdots + \underline{1}}_{n})$ for all $n \in \mathbb{N}$.

Revisiting Incompleteness

Now the precise statement of Gödel's incompleteness theorem is that the following decision problem is undecidable.

Problem (true arithmetic = "first-order theory of the natural numbers")

Input: *an* $\mathcal{L}_{arith}$*-sentence* φ

Question: *Is* $\mathbb{N} \models \varphi$*?*

Thus we have to focus on "weaker" theories in order to develop a deciding algorithm.

Revisiting Incompleteness

Now the precise statement of Gödel's incompleteness theorem is that the following decision problem is undecidable.

Problem (true arithmetic = "first-order theory of the natural numbers")

Input: *an* $\mathcal{L}_{arith}$*-sentence* φ

Question: *Is* $\mathbb{N} \models \varphi$*?*

Thus we have to focus on "weaker" theories in order to develop a deciding algorithm.

Modifying Our Goal

We are going to construct an algorithm deciding the following problem.

Problem (Presburger arithmetic)

Input: *an $\mathcal{L}_{Pres}$-sentence φ*

Question: *Is $\mathbb{N} \models \varphi$?*

Remark

As a corollary of existence of a deciding algorithm, multiplication $z = x \times y$ is not definable as an $\mathcal{L}_{\text{Pres}}$-formula $\varphi(x, y, z)$.

Modifying Our Goal

We are going to construct an algorithm deciding the following problem.

Problem (Presburger arithmetic)

Input: *an $\mathcal{L}_{Pres}$-sentence φ*

Question: *Is $\mathbb{N} \models \varphi$?*

Remark

As a corollary of existence of a deciding algorithm, multiplication $z = x \times y$ is not definable as an $\mathcal{L}_{\text{Pres}}$-formula $\varphi(x, y, z)$.

Table of Contents

The Aim of This Section

We need some techniques from automata theory to construct an algorithm.

So we are going to recall the basics of automata theory quickly.

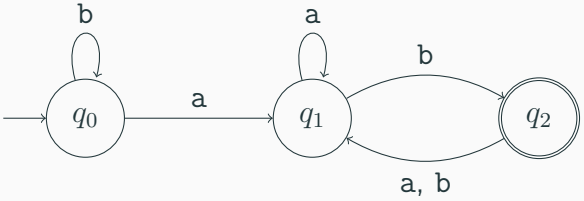
What is a finite automaton?

A finite automaton is a simple mathematical model of the computation, whose storage is extremely limited. Roughly speaking,

$$\text{finite automaton} \approx \text{computer} - \text{its storage}.$$

Finite automata can memorize only finite and fixed amount of information as its current state.

A finite automaton can be visualized as a state diagram like:



Strings

Definition

For a finite set Σ, we write Σ^* for the set of strings over Σ.

Example

If $\Sigma = \{\mathtt{a}, \mathtt{b}\}$, then

$$\Sigma^* = \{\varepsilon, \mathtt{a}, \mathtt{b}, \mathtt{aa}, \mathtt{ab}, \mathtt{ba}, \mathtt{bb}, \mathtt{aaa}, \dots\}$$

where ε is the empty string, whose length is zero.

Formal Definition of DFA

Definition

A deterministic finite automata (DFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ consisting of the following data.

1. a finite set Q of the states,
2. a finite set Σ of the alphabet.
3. transition function $\delta \colon Q \times \Sigma \to Q$,
4. the start state $q_0 \in Q$,
5. a subset $F \subseteq Q$ of the accept states.

Formal Definition of NFA

Definition

An nondeterministic finite automata (NFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ consisting of the following data.

1. a finite set Q of the states,
2. a finite set Σ of the alphabet.
3. transition function $\delta \colon Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$,
4. the start state $q_0 \in Q$,
5. a subset $F \subseteq Q$ of the accept states.

Computation by DFA

Definition

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA. A string $w \in \Sigma^*$ is accepted by M if there are

- a partition $w = w_1 w_2 \cdots w_n$ such that $w_i \in \Sigma$ and
- a finite sequence $r_0, r_1, \ldots, r_n \in Q$ of states

such that

1. $r_0 = q_0$,
2. $\delta(r_i, w_{i+1}) = r_{i+1}$ for $i = 0, 1, \ldots, n-1$,
3. $r_n \in F$.

Otherwise w is rejected by M.

Computation by NFA

Definition

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA. A string $w \in \Sigma^*$ is accepted by N if there are

- a partition $w = w_1 w_2 \cdots w_n$ such that $w_i \in \Sigma \cup \{\varepsilon\}$ and
- a finite sequence $r_0, r_1, \ldots, r_n \in Q$ of states

such that

1. $r_0 = q_0$,
2. $\delta(r_i, w_{i+1}) \ni r_{i+1}$ for $i = 0, 1, \ldots, n-1$,
3. $r_n \in F$.

Otherwise w is rejected by M.

Regular Languages

Definition (languages)

Let Σ be a finite set. A language over Σ is a subset of Σ^*.

Definition (regular languages)

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA or an NFA. We call the set

$$L(M) = \{\, w \in \Sigma^* \mid M \text{ accepts } w \,\}$$

the language of M.

We say that a language L is recognized by M if $L = L(M)$. A language L is called regular if L is recognized by some DFA.

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

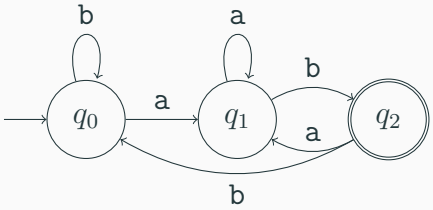


Figure 1: DFA M

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

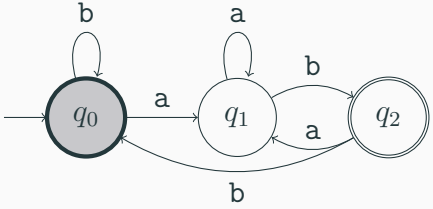


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

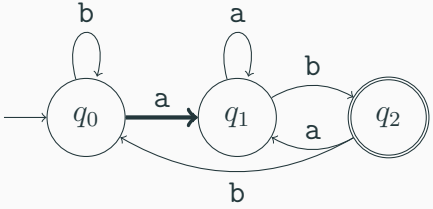


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

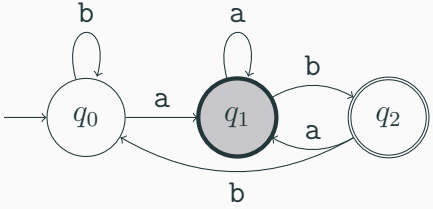


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

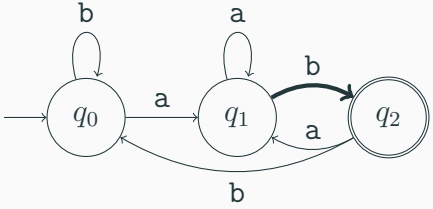


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

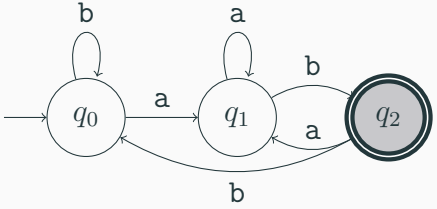


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

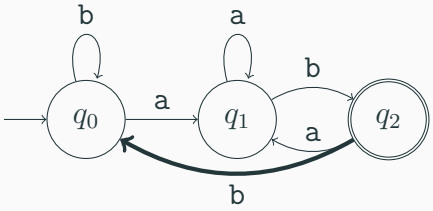


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

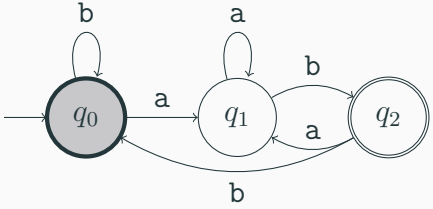


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

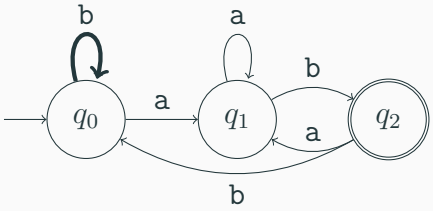


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

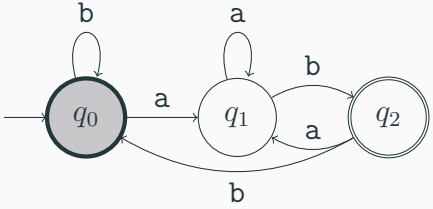


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

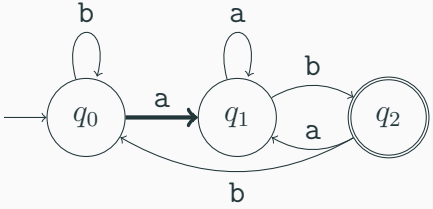


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

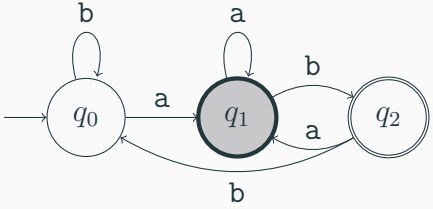


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

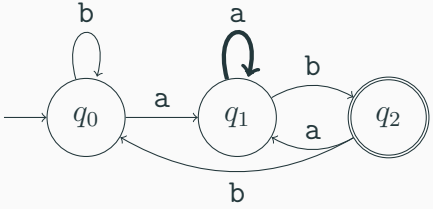


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

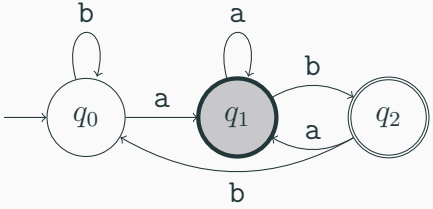


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

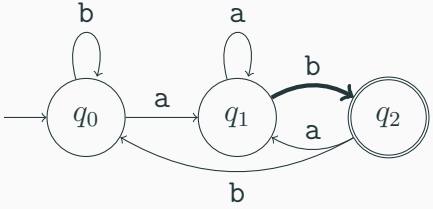


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

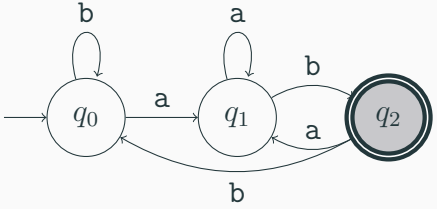


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

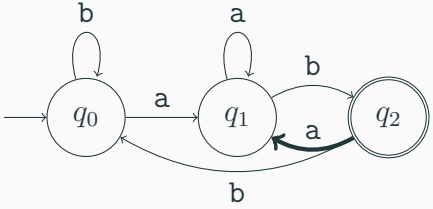


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

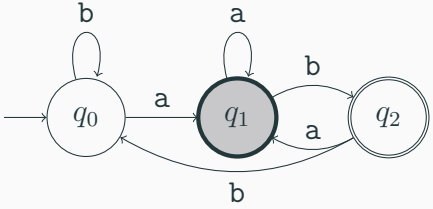


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab

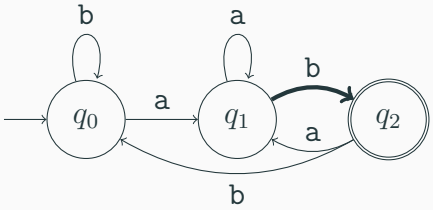


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab accepted!

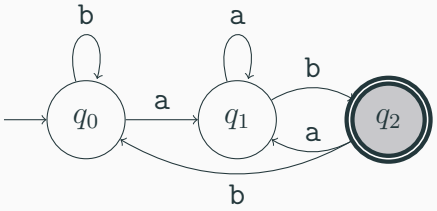


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = q_1, \delta(q_0, \mathtt{b}) = q_0$,
- $\delta(q_1, \mathtt{a}) = q_1, \delta(q_1, \mathtt{b}) = q_2$,
- $\delta(q_2, \mathtt{a}) = q_1, \delta(q_2, \mathtt{b}) = q_0$,
- $F = \{q_2\}$.

input: abbbaabab accepted!

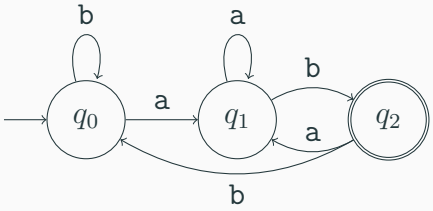


Figure 1: DFA M

If we feed a string abbbaabab to M as an input string, the computation proceeds as in Figure 1. We see that

$$L(M) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

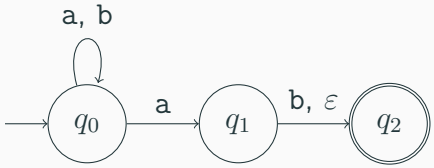


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

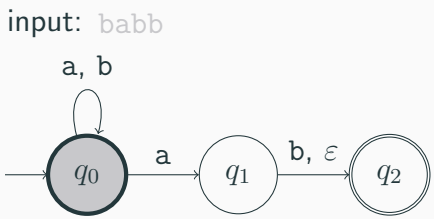


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

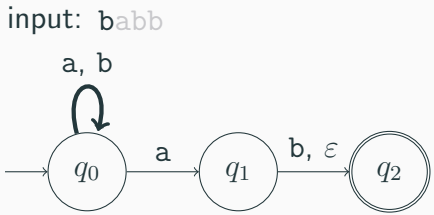


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

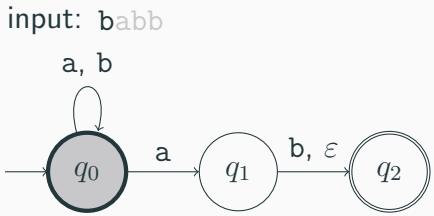


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

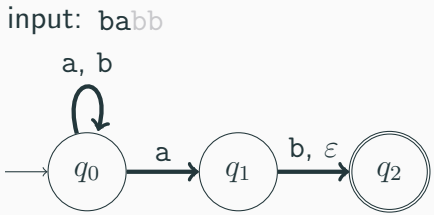


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

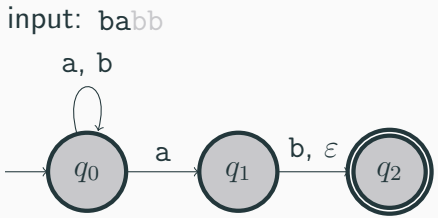


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

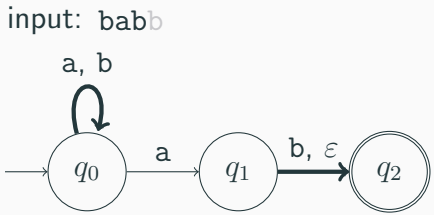


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

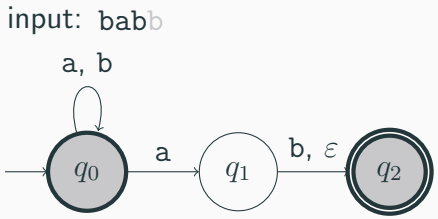


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

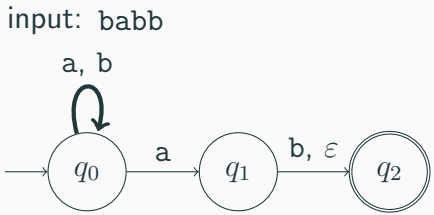


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

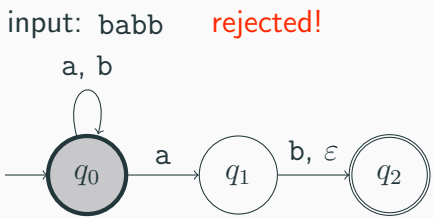


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with ab or a} \,\}.$$

An Example of NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA defined by:

- $Q = \{q_0, q_1, q_2\}$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $\delta(q_0, \mathtt{a}) = \{q_0, q_1\}, \delta(q_0, \mathtt{b}) = \{q_0\}, \delta(q_0, \varepsilon) = \emptyset$,
- $\delta(q_1, \mathtt{a}) = \emptyset, \delta(q_1, \mathtt{b}) = \{q_2\}, \delta(q_0, \varepsilon) = \{q_2\}$,
- $\delta(q_2, \mathtt{a}) = \emptyset, \delta(q_2, \mathtt{b}) = \emptyset, \delta(q_0, \varepsilon) = \emptyset$,
- $F = \{q_2\}$.

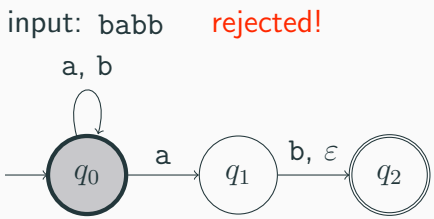


Figure 2: NFA N

If we feed a string babb to N as an input string, the computation proceeds as in Figure 2. We see that

$$L(N) = \{\, w \in \Sigma^* \mid w \text{ terminates with } \mathtt{ab} \text{ or } \mathtt{a} \,\}.$$

Subset Construction: Converting NFA into DFA

Theorem

For any NFA N, there exists a DFA M such that $L(N) = L(M)$.

Demonstration

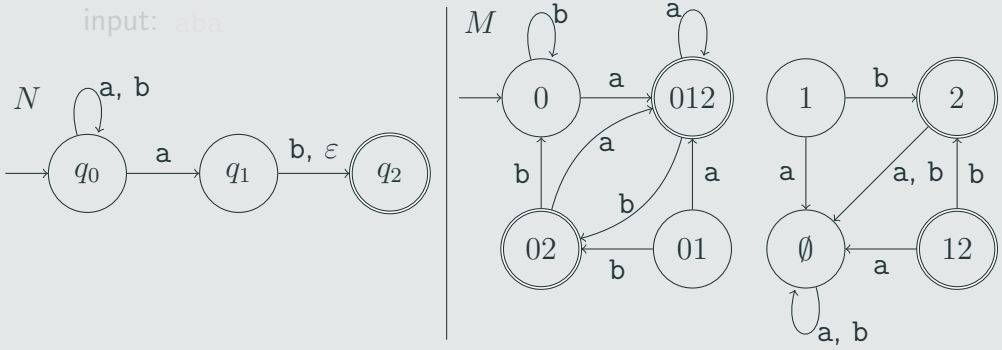
input: abb

Subset Construction: Converting NFA into DFA

Theorem

For any NFA N, there exists a DFA M such that $L(N) = L(M)$.

Demonstration

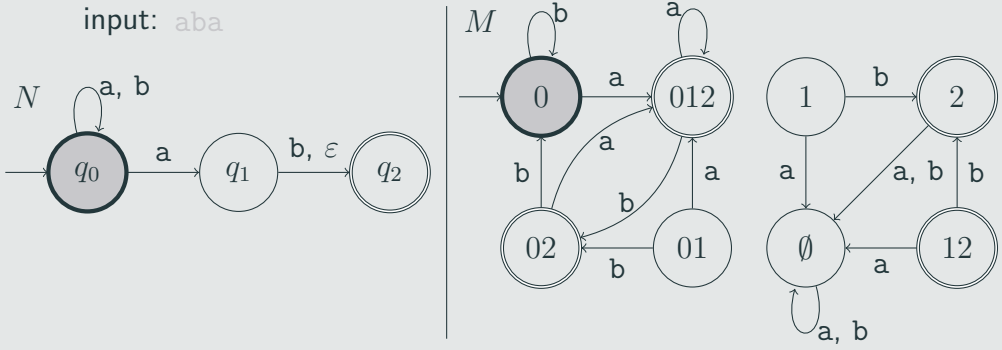


Subset Construction: Converting NFA into DFA

Theorem

For any NFA N, there exists a DFA M such that $L(N) = L(M)$.

Demonstration

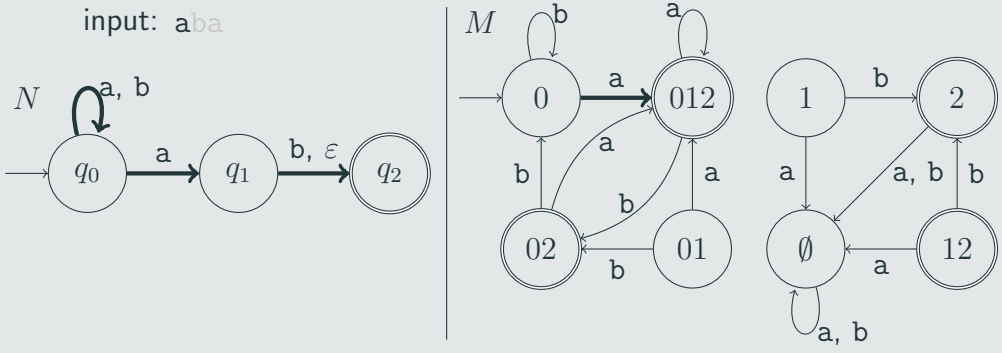


Subset Construction: Converting NFA into DFA

Theorem

For any NFA N, there exists a DFA M such that $L(N) = L(M)$.

Demonstration

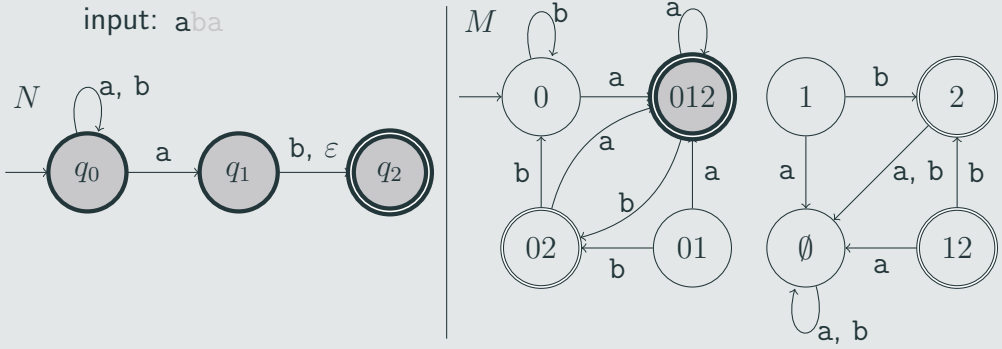


Subset Construction: Converting NFA into DFA

Theorem

For any NFA N, there exists a DFA M such that $L(N) = L(M)$.

Demonstration

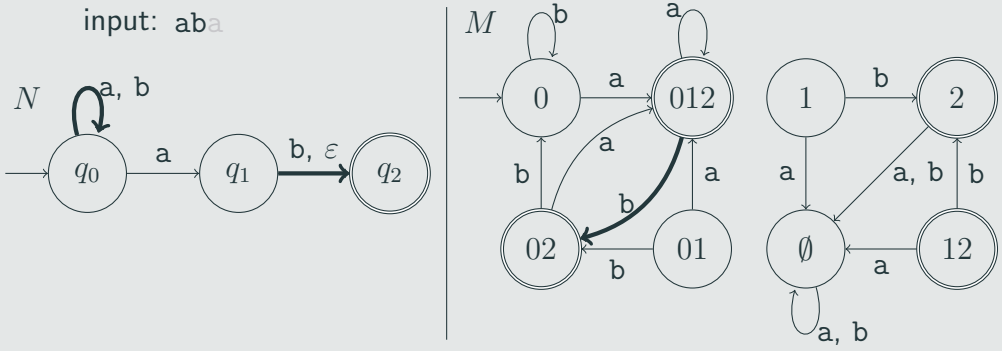


Subset Construction: Converting NFA into DFA

Theorem

For any NFA N, there exists a DFA M such that $L(N) = L(M)$.

Demonstration

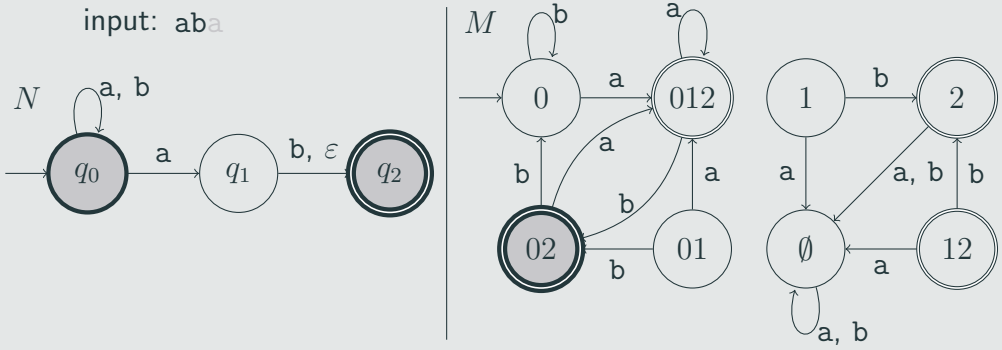


Subset Construction: Converting NFA into DFA

Theorem

For any NFA N, there exists a DFA M such that $L(N) = L(M)$.

Demonstration

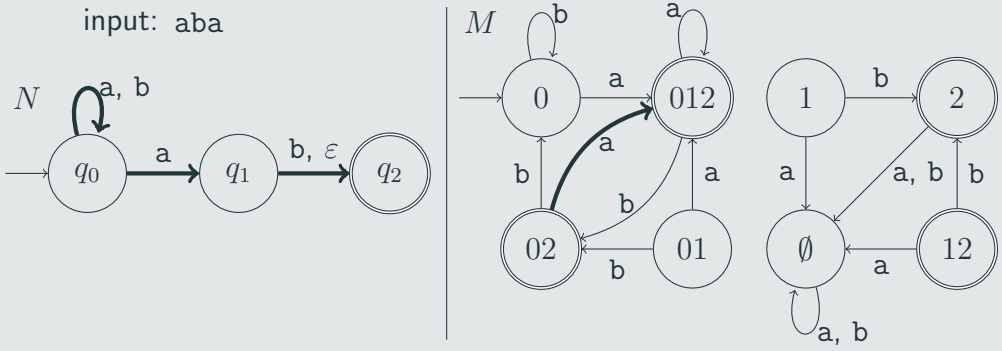


Subset Construction: Converting NFA into DFA

Theorem

For any NFA N, there exists a DFA M such that $L(N) = L(M)$.

Demonstration

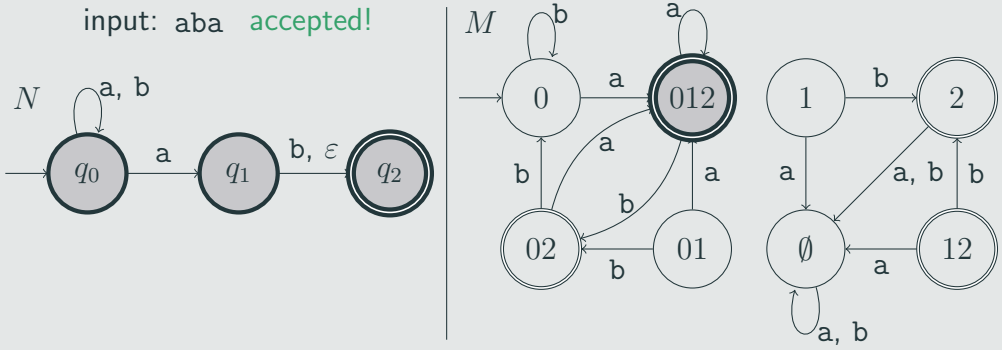


Subset Construction: Converting NFA into DFA

Theorem

For any NFA N, there exists a DFA M such that $L(N) = L(M)$.

Demonstration



Closure Properties of the Class of the Regular Languages

The class of the regular languages is closed under usual operations.

Theorem

For any regular languages $L_1, L_2 \subseteq \Sigma^*$, the following are also regular.

Complement	$\Sigma^* \setminus L_1$,
Intersection	$L_1 \cap L_2$,
Union	$L_1 \cup L_2$.

Proof Sketch.

Let $M_i = (Q_i, \Sigma, \delta_i, q_i, F_i)$ be a DFA recognizing L_i for $i = 1, 2$. Then

$$M_{\mathsf{C}} = (Q_1, \Sigma, \delta_1, q_1, Q \setminus F_1),$$
$$M_{\mathsf{I}} = (Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (q_1, q_2), F_1 \times F_2),$$
$$M_{\mathsf{U}} = (Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (q_1, q_2), (F_1 \times Q_2) \cup (Q_1 \times F_2))$$

recognizes $\Sigma^* \setminus L_1$, $L_1 \cap L_2$, $L_1 \cup L_2$, respectively. □

Closure Properties of the Class of the Regular Languages

The class of the regular languages is closed under usual operations.

Theorem

For any regular languages $L_1, L_2 \subseteq \Sigma^*$, the following are also regular.

Complement	$\Sigma^* \setminus L_1$,
Intersection	$L_1 \cap L_2$,
Union	$L_1 \cup L_2$.

Proof Sketch.

Let $M_i = (Q_i, \Sigma, \delta_i, q_i, F_i)$ be a DFA recognizing L_i for $i = 1, 2$. Then

$$M_{\mathsf{C}} = (Q_1, \Sigma, \delta_1, q_1, Q \setminus F_1),$$
$$M_{\mathsf{I}} = (Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (q_1, q_2), F_1 \times F_2),$$
$$M_{\mathsf{U}} = (Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (q_1, q_2), (F_1 \times Q_2) \cup (Q_1 \times F_2))$$

recognizes $\Sigma^* \setminus L_1$, $L_1 \cap L_2$, $L_1 \cup L_2$, respectively. □

Table of Contents

The Aim of This Section

Now let's build a deciding algorithm for Presburger arithmetic!

Proof (1/10)

Let φ be an $\mathcal{L}_{\text{Pres}}$-sentence. We want to examine the truth value of φ.

First we decompose φ into "simple" parts.

Proof (2/10) — Step 1: Eliminating the Inequality Sign $<$

It is easy to see that

$$t_1 < t_2 \iff \exists x((t_1 + x) + \underline{1}) = t_2.$$

Thus we may assume that φ contians no inequality sign $<$. In other words, φ consists of equations.

Proof (3/10) — Step 2: Decomposing Equations

We can replace each of the equations occurring in φ into a combinations of some equations of the form $(\alpha + \beta) = \gamma$ where α, β, γ are variable symbols or $\underline{0}, \underline{1}$.

Example

One can rewrite the equation

$$(((x + y) + \underline{1}) + x) = (z + z)$$

into

$$\exists u \exists v \exists w ((((x + y) = u \wedge (u + \underline{1}) = v) \wedge (v + x) = w) \wedge (z + z) = w)$$

at the cost of introducing new variable symbols u, v and w.

Proof (4/10) — Step 3: Converting into Prenex Normal Form

Any first-order formula can be converted into the prenex normal form by factoring out $\exists$ and $\forall$.

Example

One can rewrite the $\mathcal{L}_{\mathsf{Pres}}$-sentence

$$\exists x(\exists y(y+y) = x \wedge \forall y \neg x = (y + \underline{1}))$$

into the prenex normal form

$$\exists x \exists y \forall z((y+y) = x \wedge \neg x = (z + \underline{1}))$$

by changing the name of one of the variables y into z.

Thus we may assume that φ is of the form

$$\mathsf{Q}_1 x_1 \mathsf{Q}_2 x_2 \cdots \mathsf{Q}_n x_n \psi(x_1, x_2, \ldots, x_n)$$

where $\psi(x_1, x_2, \ldots, x_n)$ is a Boolean combination of equations of the form $(\alpha + \beta) = \gamma$ and Q_i is either $\exists$ or $\forall$.

Proof (4/10) — Step 3: Converting into Prenex Normal Form

Any first-order formula can be converted into the prenex normal form by factoring out $\exists$ and $\forall$.

Example

One can rewrite the $\mathcal{L}_{\mathsf{Pres}}$-sentence

$$\exists x(\exists y(y+y) = x \wedge \forall y \neg x = (y + \underline{1}))$$

into the prenex normal form

$$\exists x \exists y \forall z((y+y) = x \wedge \neg x = (z + \underline{1}))$$

by changing the name of one of the variables y into z.

Thus we may assume that φ is of the form

$$\mathsf{Q}_1 x_1 \mathsf{Q}_2 x_2 \cdots \mathsf{Q}_n x_n \psi(x_1, x_2, \ldots, x_n)$$

where $\psi(x_1, x_2, \ldots, x_n)$ is a Boolean combination of equations of the form $(\alpha + \beta) = \gamma$ and Q_i is either $\exists$ or $\forall$.

Proof (5/10)

Let ψ_i be a subformula of φ for $i = 0, 1, \ldots, n$ defined by

$$\begin{aligned}
\psi_0 &\equiv \mathsf{Q}_1 x_1 \mathsf{Q}_2 x_2 \cdots \mathsf{Q}_n x_n \psi(x_1, \ldots, x_n) \ (\equiv \varphi), \\
\psi_1(x_1) &\equiv \mathsf{Q}_2 x_2 \cdots \mathsf{Q}_n x_n \psi(x_1, \ldots, x_n), \\
&\cdots, \\
\psi_i(x_1, \ldots, x_i) &\equiv \mathsf{Q}_{i+1} x_{i+1} \cdots \mathsf{Q}_n x_n \psi(x_1, \ldots, x_n), \\
&\cdots, \\
\psi_{n-1}(x_1, \ldots, x_{n-1}) &\equiv \mathsf{Q}_n x_n \psi(x_1, \ldots, x_n), \\
\psi_n(x_1, \ldots, x_{n-1}, x_n) &\equiv \psi(x_1, \ldots, x_n).
\end{aligned}$$

Proof Idea

For each $i = n, n-1, \ldots, 1, 0$, we are going to construct a finite automaton M_i, which recognizes the values $(a_1, \ldots, a_i) \in \mathbb{N}^i$ satisfying $\mathbb{N} \models \psi_i(a_1, \ldots, a_i)$.

Then we can obtain the truth value of φ by testing whether M_0 accepts ε!

Proof (5/10)

Let ψ_i be a subformula of φ for $i = 0, 1, \ldots, n$ defined by

$$\begin{aligned}
\psi_0 &\equiv \mathsf{Q}_1 x_1 \mathsf{Q}_2 x_2 \cdots \mathsf{Q}_n x_n \psi(x_1, \ldots, x_n) \ (\equiv \varphi), \\
\psi_1(x_1) &\equiv \mathsf{Q}_2 x_2 \cdots \mathsf{Q}_n x_n \psi(x_1, \ldots, x_n), \\
&\ldots, \\
\psi_i(x_1, \ldots, x_i) &\equiv \mathsf{Q}_{i+1} x_{i+1} \cdots \mathsf{Q}_n x_n \psi(x_1, \ldots, x_n), \\
&\ldots, \\
\psi_{n-1}(x_1, \ldots, x_{n-1}) &\equiv \mathsf{Q}_n x_n \psi(x_1, \ldots, x_n), \\
\psi_n(x_1, \ldots, x_{n-1}, x_n) &\equiv \psi(x_1, \ldots, x_n).
\end{aligned}$$

Proof Idea

For each $i = n, n-1, \ldots, 1, 0$, we are going to construct a finite automaton M_i, which recognizes the values $(a_1, \ldots, a_i) \in \mathbb{N}^i$ satisfying $\mathbb{N} \models \psi_i(a_1, \ldots, a_i)$.

Then we can obtain the truth value of φ by testing whether M_0 accepts ε!

Proof (6/10) — How to Feed the Number to Automata

For $i = 1, 2, \ldots, n$, let

$$\Sigma_i = \left\{ \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 1 \end{bmatrix}, \ldots \begin{bmatrix} 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix} \right\}$$

be the alphabet for M_i, which has 2^i elements. We define $\Sigma_0 = \emptyset$ and $\Sigma_0^* = \{\varepsilon\}$.

In order to know the truth value of $\psi_i(a_1, \ldots, a_i)$, we just need to feed M_i with the binary expansions of $a_1, \ldots, a_i$ from the lowest digit.

Example

Suppose we want to check $\psi_3(3, 4, 13)$. Since $3 = (11)_2$, $4 = (100)_2$ and $13 = (1101)_2$, we will feed M_3 with

$$\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Proof (6/10) — How to Feed the Number to Automata

For $i = 1, 2, \ldots, n$, let

$$\Sigma_i = \left\{ \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 1 \end{bmatrix}, \ldots \begin{bmatrix} 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix} \right\}$$

be the alphabet for M_i, which has 2^i elements. We define $\Sigma_0 = \emptyset$ and $\Sigma_0^* = \{\varepsilon\}$.

In order to know the truth value of $\psi_i(a_1, \ldots, a_i)$, we just need to feed M_i with the binary expansions of $a_1, \ldots, a_i$ from the lowest digit.

Example

Suppose we want to check $\psi_3(3, 4, 13)$. Since $3 = (11)_2$, $4 = (100)_2$ and $13 = (1101)_2$, we will feed M_3 with

$$\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Proof (7/10) — Computing Additions with DFA (1/2)

First we construct M_n that recognizes $\psi_n(a_1, \ldots, a_n)$.
It suffices to check the validity of formulas of the form $\alpha + \beta = \gamma$ thanks to the closure properties of regular languages.
It is the most important case that $x_i + x_j = x_k$ with $i \neq j \neq k \neq i$.

Figure 3: DFA for $a_1 + a_2 = a_3$

Proof (7/10) — Computing Additions with DFA (1/2)

First we construct M_n that recognizes $\psi_n(a_1, \ldots, a_n)$.
It suffices to check the validity of formulas of the form $\alpha + \beta = \gamma$ thanks to the closure properties of regular languages.
It is the most important case that $x_i + x_j = x_k$ with $i \neq j \neq k \neq i$.

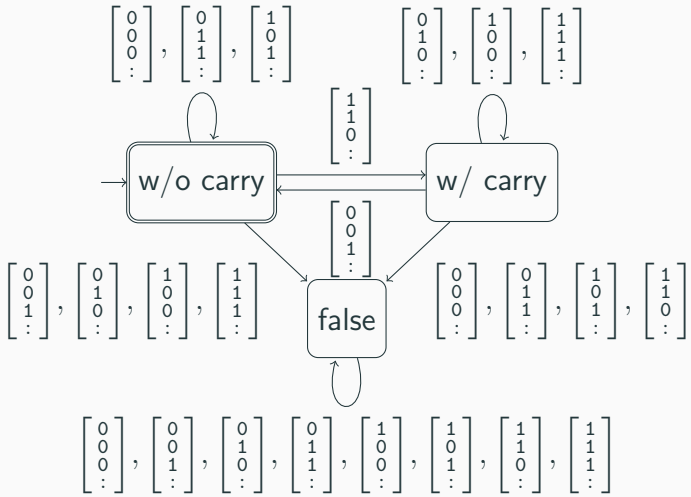


Figure 3: DFA for $a_1 + a_2 = a_3$

Proof (8/10) — Computing Additions with DFA (2/2)

In the other cases,

$\alpha, \beta, \gamma \in \{\underline{0}, \underline{1}\}$	$\Rightarrow$ Accept or reject all of the inputs
$(x_i + \underline{0}) = x_i$	$\Rightarrow$ Accept all of the inputs
$(x_i + x_i) = \underline{1}$ or $(x_i + \underline{1}) = x_i$	$\Rightarrow$ Reject all of the inputs
$(\underline{0} + \underline{0}) = x_i$ or $(x_i + x_i) = \underline{0}$ or $(x_i + x_i) = x_i$ or $(x_i + x_j) = x_j$ $(i \neq j)$	$\Rightarrow$ Check if $a_i = 0$
$(x_i + \underline{0}) = x_j$ $(i \neq j)$	$\Rightarrow$ Check if $a_i = a_j$
$(x_i + x_j) = \underline{0}$ $(i \neq j)$	$\Rightarrow$ Check if $a_i = a_j = 0$
$(\underline{0} + \underline{1}) = x_i$	$\Rightarrow$ Check if $a_i = 1$
$(\underline{1} + \underline{1}) = x_i$	$\Rightarrow$ Check if $a_i = 2$
$(x_i + x_i) = x_j$ $(i \neq j)$	$\Rightarrow$ Similar to the previous page
$(x_i + x_j) = \underline{1}$ $(i \neq j)$ or $(x_i + \underline{1}) = x_j$ $(i \neq j)$	$\Rightarrow$ Similar to the previous page

Proof (9/10) — Manipulating the Quantifier $\exists$

We construct an NFA N_{n-1} based on the DFA M_n. Since $\forall x$ is equivalent to $\neg\exists x\neg$, we may assume that $\psi_{n-1}(a_1, \ldots, a_{n-1}) \equiv \exists x_n \psi(a_1, \ldots, a_{n-1}, x_n)$.

The new NFA N_{n-1} nondeterministically guess the value of x_n. Namely, N_{n-1} can be obtained by truncating the n-th row of each transition in M_n.

Figure 4: DFA M_3 for $(a_1 + a_3) = a_2$

Proof (9/10) — Manipulating the Quantifier $\exists$

We construct an NFA N_{n-1} based on the DFA M_n. Since $\forall x$ is equivalent to $\neg\exists x\neg$, we may assume that $\psi_{n-1}(a_1,\ldots,a_{n-1}) \equiv \exists x_n \psi(a_1,\ldots,a_{n-1},x_n)$.

The new NFA N_{n-1} nondeterministically guess the value of x_n. Namely, N_{n-1} can be obtained by truncating the n-th row of each transition in M_n.

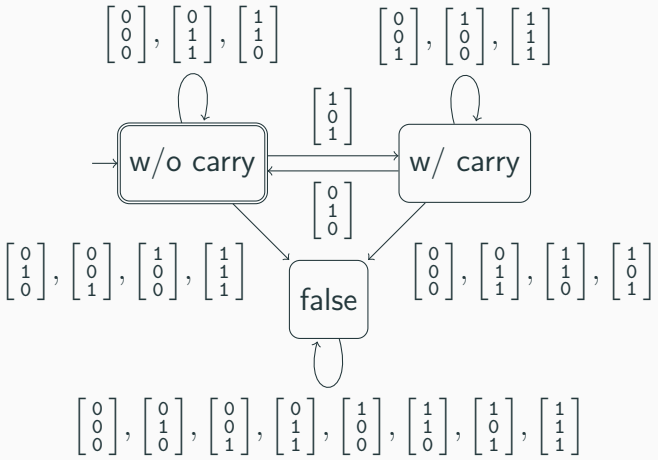


Figure 4: DFA M_3 for $(a_1 + a_3) = a_2$

Proof (9/10) — Manipulating the Quantifier $\exists$

We construct an NFA N_{n-1} based on the DFA M_n. Since $\forall x$ is equivalent to $\neg\exists x\neg$, we may assume that $\psi_{n-1}(a_1, \ldots, a_{n-1}) \equiv \exists x_n \psi(a_1, \ldots, a_{n-1}, x_n)$.

The new NFA N_{n-1} nondeterministically guess the value of x_n. Namely, N_{n-1} can be obtained by truncating the n-th row of each transition in M_n.

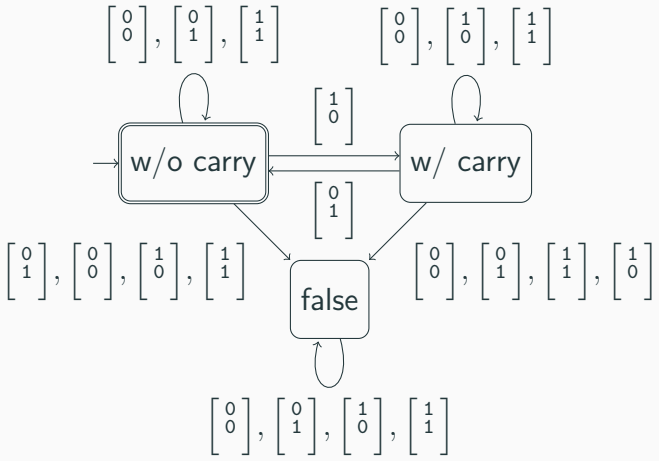


Figure 4: NFA N_2 for $\exists x_3(a_1 + x_3) = a_2$

Proof (10/10) — Summary

Now we can examine the validity of φ in the following way.

1. Eliminate the inequality signs $<$ in φ.
2. Replace each equation in φ with combinations of equations of the form $(\alpha + \beta) = \gamma$.
3. Convert φ into the prenex normal form.
4. Construct the DFA M_n for $\psi_n(a_1, \ldots, a_n)$.
5. Construct the NFA N_{n-1} from M_n by truncating the n-th row.
6. Construct the DFA M_{n-1} from N_{n-1} by the subset construction method.
7. Construct M_0 by repeating this process.
8. Check whether M_0 accepts ε.

Q.E.D.

Proof (10/10) — Summary

Now we can examine the validity of φ in the following way.

1. Eliminate the inequality signs $<$ in φ.
2. Replace each equation in φ with combinations of equations of the form $(\alpha + \beta) = \gamma$.
3. Convert φ into the prenex normal form.
4. Construct the DFA M_n for $\psi_n(a_1, \ldots, a_n)$.
5. Construct the NFA N_{n-1} from M_n by truncating the n-th row.
6. Construct the DFA M_{n-1} from N_{n-1} by the subset construction method.
7. Construct M_0 by repeating this process.
8. Check whether M_0 accepts ε.

Q.E.D.

Reference

M. Sipser, *Introduction to the Theory of Computation*, Third Edition, Cengage Learning, 2012.