離散最適化基礎論 第 11 回
木幅と論理：アルゴリズム設計

岡本 吉央
okamotoy@uec.ac.jp

電気通信大学

2017 年 1 月 27 日

最終更新：2017 年 1 月 30 日　14:44

スケジュール 前半

1. 離散最適化における木分解の役割 (10/7)
- ⋆ 休講 (国内出張) (10/14)
2. 木に対するアルゴリズム設計 (10/23)
3. 道幅と道分解 (10/30)
4. 道分解を用いたアルゴリズム設計 (11/4)
- ⋆ 休講 (海外出張) (11/11)
5. 木分解と木幅 (11/18)
- ⋆ 休講 (調布祭) (11/25)
6. 木幅の性質 (12/2)

スケジュール 後半 (予定)

7 木分解を用いたアルゴリズム設計 (12/9)
8 木分解を用いたアルゴリズム設計：連結性 (12/16)
⋆ 休講 (天皇誕生日) (12/23)
⋆ 冬季休業 (12/30)
9 木幅と論理：単項二階論理 (1/6)
⋆ 休講 (センター試験準備) (1/13)
10 木幅と論理：オートマトン (1/20)
11 木幅と論理：アルゴリズム設計 (1/27)
12 木分解構成アルゴリズム：準備 (2/3)
13 木分解構成アルゴリズム (2/10)
⋆ 期末試験 (2/17)

注意：予定の変更もありうる

主題

離散最適化のトピックの１つとして
グラフの木分解を取り上げ，

- 木分解とは何か？
- 木分解がなぜ役に立つのか？
- 木分解がどう役に立つのか？

について，数理的側面と計算的側面の双方を意識して講義する

なぜ講義で取り扱う？

- 「離散最適化の神髄」だから

木幅 — 「木っぽさ」を表す尺度

この講義のキーワード (と荒っぽい説明)

グラフの木幅	グラフの「木っぽさ」を表す尺度 (の 1 つ)
グラフの木分解	グラフを「木っぽく」表した構造
動的計画法	木分解上の効率的アルゴリズム
オートマトン	動的計画法に基づくアルゴリズムの解釈
Courcelle の定理	上記と論理学に基づく『メタアルゴリズム』

木幅と木分解の面白さ

次の主題が有機的に結びつく面白い話題

- グラフ
- アルゴリズム
- オートマトン
- 論理 (特に，有限モデル理論)

ポイント

効率的アルゴリズムが設計できる背景に「美しい数理構造」がある

この講義では，その一端に触れたい

今日の目標

- Büchi の定理の証明の残り (アルゴリズム)
- Courcelle の定理の証明の雰囲気

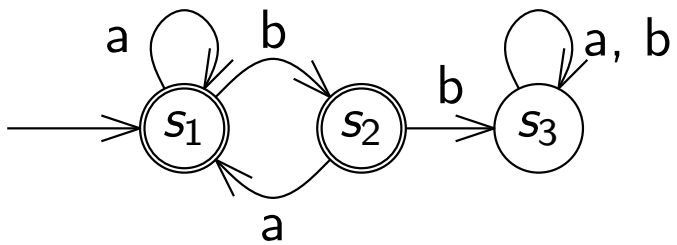
目次

非決定性有限オートマトンとは？

次の 5 つから構成される $\mathfrak{A}$

- 有限集合 S： 状態の集合
- 有限集合 Σ： アルファベット
- $s_I \in S$： 初期状態
- $\Delta \subseteq S \times \Sigma \times S$： 遷移関係
- $F \subseteq S$： 受理状態の集合

$\mathfrak{A} = (S, \Sigma, s_I, \Delta, F)$ と書くこともある

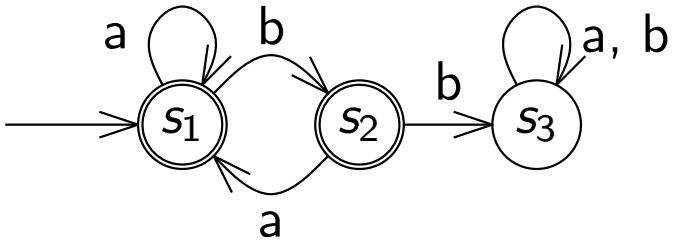


$$S = \{s_1, s_2, s_3\}, \Sigma = \{\mathrm{a}, \mathrm{b}\}, s_I = s_1, F = \{s_1, s_2\}$$

非決定性有限オートマトン (続)

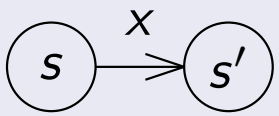
遷移関係 $\Delta \subseteq S \times \Sigma \times S$

$$\Delta = \{(s_1, \mathrm{a}, s_1), (s_1, \mathrm{b}, s_2), (s_2, \mathrm{a}, s_1), (s_2, \mathrm{b}, s_3), (s_3, \mathrm{a}, s_3), (s_3, \mathrm{b}, s_3)\}$$



遷移関係の見方

$$(s, x, s') \in \Delta \quad \Leftrightarrow$$



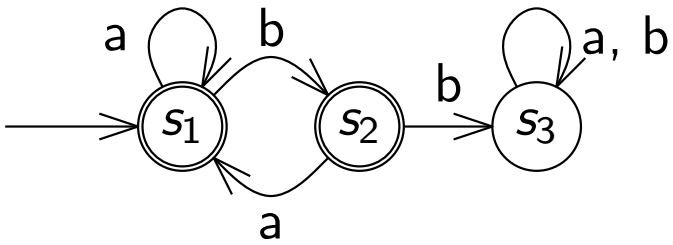
非決定性有限オートマトン $\mathfrak{A} = (S, \Sigma, s_I, \Delta, F)$,
文字列 $w = x_1 x_2 \ldots x_n \in \Sigma^*$

オートマトンによる文字列の受理

$\mathfrak{A}$ が w を受理するとは，次を満たす $s_0, s_1, \ldots, s_n$ が存在すること

- $s_0 = s_I$ (初期状態)
- 任意の $i \in \{1, \ldots, n\}$ に対して，$(s_{i-1}, x_i, s_i) \in \Delta$ (遷移)
- $s_n \in F$ (受理状態)

受理しないとき，棄却するという



$w =$ ababa のとき，
$(s_1, \mathrm{a}, s_1), (s_1, \mathrm{b}, s_2), (s_2, \mathrm{a}, s_1), (s_1, \mathrm{b}, s_2), (s_2, \mathrm{a}, s_1) \in \Delta$

アルファベット Σ

言語 (language) とは？

Σ 上の言語 L とは，Σ 上の文字列の集合

$$L \subseteq \Sigma^*$$

正則言語 (regular language) とは？

言語 L が正則であるとは，ある非決定性有限オートマトン $\mathfrak{A}$ が存在して

$$L = \{w \mid \mathfrak{A} \text{ は } w \text{ を受理する}\}$$

注意

- 「正則言語」は「正規言語」とも呼ばれる
- 上に挙げた正則言語の定義は本来定理であり，普通は別の形 (文法) で定義する

詳細は『オートマトン理論』,『形式言語理論』などを参照

アルファベット Σ，文字列 $w = w_1 w_2 \dots w_n \in \Sigma^*$

文字列に対する単項二階論理式

変数

- 位置 $i \in [n]$
- 位置の部分集合 $X \subseteq [n]$

論理結合子

- $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$

量化子

- $\forall$, $\exists$

定数

- Σ の要素

述語

- $i, j \in [n]$ に対して，「$i = j$」
- $i \in [n]$, $X \subseteq [n]$ に対して，「$i \in S$」
- $X_1, X_2 \subseteq [n]$ に対して，「$X_1 \subseteq X_2$」
- $i \in [n], \mathrm{a} \in \Sigma$ に対して，「$w_i = \mathrm{a}$」
- $i, j \in [n]$ に対して，「$i \leq j$」

ただし，$[n] = \{1, 2, \dots, n\}$

これらを用いて記述できる論理式が単項二階論理式

例

$\Sigma = \{\mathsf{a}, \mathsf{b}\}$ で，$w \in \Sigma^*$ が次を満たすとする

b が 2 つ続かない

- このとき，

$$\begin{aligned} \phi \quad = \quad & \forall\, i\, (w_i = \mathsf{b} \rightarrow \\ & \neg\exists\, j\, (w_j = \mathsf{b} \wedge i \leq j \wedge i \neq j \wedge \forall\, k\, (k \leq i \vee j \leq k))) \end{aligned}$$

とすると，$w \models \phi$

- 逆に，$w \models \phi$ ならば，w において，b が 2 つ続かない

単項二階論理式が定義する言語

アルファベット Σ，単項二階論理式 ϕ

単項二階論理式が定義する言語とは？

ϕ が定義する言語とは，次の言語

$$\{w \in \Sigma^* \mid w \models \phi\}$$

先ほどの例：$\Sigma = \{\mathrm{a}, \mathrm{b}\}$ で，

$$\begin{aligned} L &= \{w \in \Sigma^* \mid w \text{ において b が 2 つ続かない }\} \\ \phi &= \forall\, i\, (w_i = \mathrm{b} \rightarrow \\ &\qquad \neg\exists\, j\, (w_j = \mathrm{b} \wedge i \leq j \wedge i \neq j \wedge \forall\, k\, (k \leq i \vee j \leq k))) \end{aligned}$$

とすると，L は ϕ が定義する言語となる

Büchi の定理はオートマトンと単項二階論理をつなぐ

Büchi の定理

アルファベット Σ 上の言語 $L \subseteq \Sigma^*$ に対して

L が正則 $\Leftrightarrow$ ある単項二階論理式 ϕ が存在して，ϕ が L を定義する

先ほどの例：$\Sigma = \{\mathsf{a}, \mathsf{b}\}$ で，

$$
\begin{aligned}
L &= \{w \in \Sigma^* \mid w \text{ において } \mathsf{b} \text{ が } 2 \text{ つ続かない } \} \\
\phi &= \forall\, i\, (w_i = \mathsf{b} \rightarrow \\
&\qquad \neg\exists\, j\, (w_j = \mathsf{b} \wedge i \leq j \wedge i \neq j \wedge \forall\, k\, (k \leq i \vee j \leq k)))
\end{aligned}
$$

とすると，L は ϕ が定義する言語となり，L は正則だった

Büchi の定理はオートマトンと単項二階論理をつなぐ

Büchi の定理

アルファベット Σ 上の言語 $L \subseteq \Sigma^*$ に対して

L が正則 $\Leftrightarrow$ ある単項二階論理式 ϕ が存在して，ϕ が L を定義する

証明の方針：

- 「$\Rightarrow$」L を受理するオートマトン $\mathfrak{A}$ から ϕ を作る
- 「$\Leftarrow$」ϕ から L を受理するオートマトン $\mathfrak{A}$ を作る

前回「$\Rightarrow$」の証明を行ったので，今回は「$\Leftarrow$」の証明を行う

Büchi の定理はオートマトンと単項二階論理をつなぐ

Büchi の定理

アルファベット Σ 上の言語 $L \subseteq \Sigma^*$ に対して

L が正則 $\Leftrightarrow$ ある単項二階論理式 ϕ が存在して，ϕ が L を定義する

Büchi の定理の「$\Leftarrow$」の証明の帰結

与えられた単項二階論理式 ϕ に対して，
文字列 w が ϕ の定義する言語 L の要素かどうか判定するアルゴリズムで
$O(f(|\phi|) + |w|)$ 時間で動くものが存在する

- $f(\cdot)$ はある関数，$|\phi|$ は ϕ の長さ，$|w|$ は w の長さ
- これが Courcelle の定理の証明に効いてくる

目次

正則言語 (regular language) とは？

言語 L が正則であるとは，ある非決定性有限オートマトン $\mathfrak{A}$ が存在して

$$L = \{w \mid \mathfrak{A} \text{ は } w \text{ を受理する}\}$$

考えたいこと

正則言語 $L_1, L_2 \subseteq \Sigma^*$ から別の言語を作る

- $L_1 \cap L_2$ (共通部分)
- $L_1 \cup L_2$ (合併)
- $\overline{L_1} = \Sigma^* - L_1$ (補集合)
- など

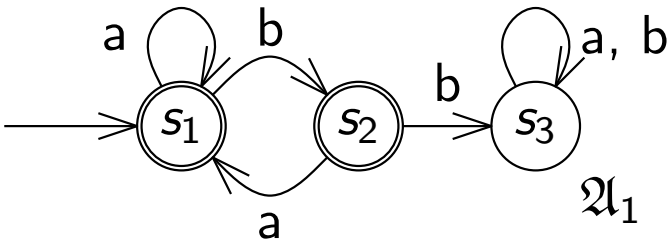
質問：これらは正則言語なのか？

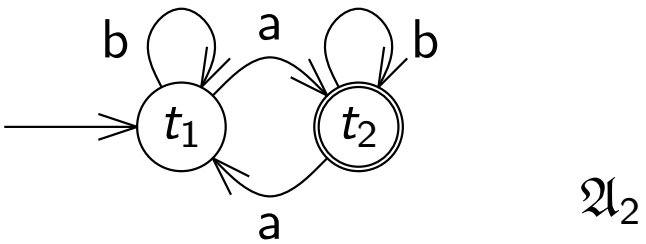
正則言語の性質 1 (証明は省略)

正則言語 $L_1, L_2 \subseteq \Sigma^*$ に対して，$L_1 \cap L_2 \subseteq \Sigma^*$ も正則言語

証明の雰囲気：直積オートマトンと呼ばれる構成法

- L_1, L_2 を受理する有限オートマトン $\mathfrak{A}_1, \mathfrak{A}_2$ から
  $L_1 \cap L_2$ を受理する有限オートマトンを構成すればよい



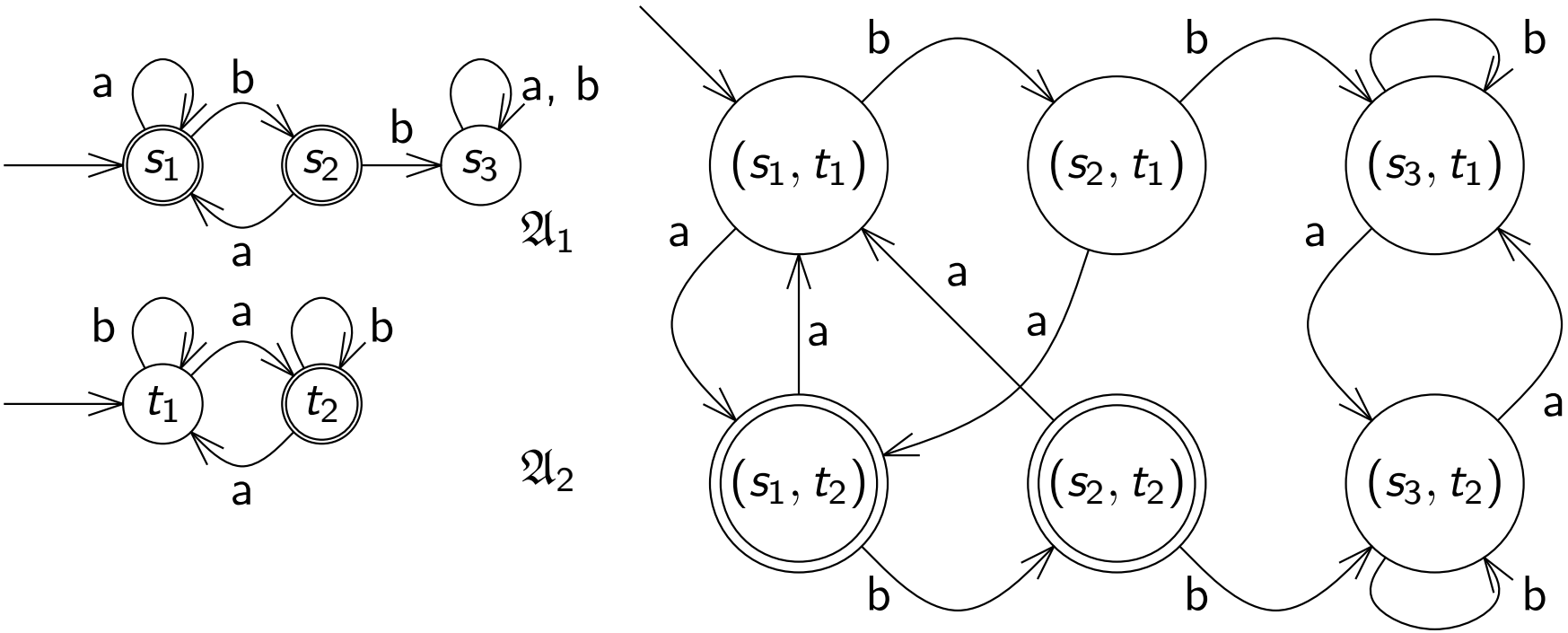


正則言語の性質 1 (証明は省略)

正則言語 $L_1, L_2 \subseteq \Sigma^*$ に対して，$L_1 \cap L_2 \subseteq \Sigma^*$ も正則言語

証明の雰囲気：直積オートマトンと呼ばれる構成法

- L_1, L_2 を受理する有限オートマトン $\mathfrak{A}_1, \mathfrak{A}_2$ から
  $L_1 \cap L_2$ を受理する有限オートマトンを構成すればよい

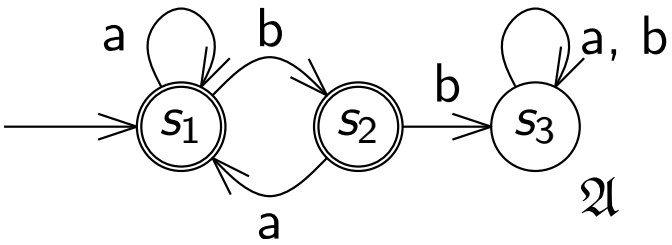


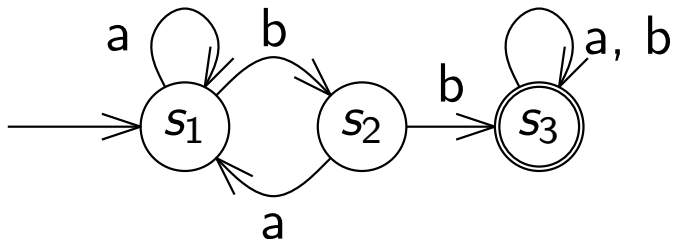
正則言語の性質 2 (証明は省略)

正則言語 $L \subseteq \Sigma^*$ に対して，$\Sigma^* - L$ も正則言語

証明の雰囲気 (この図は決定性オートマトンの場合)

- L を受理する有限オートマトン $\mathfrak{A}$ から
  $\Sigma^* - L$ を受理する有限オートマトンを構成すればよい



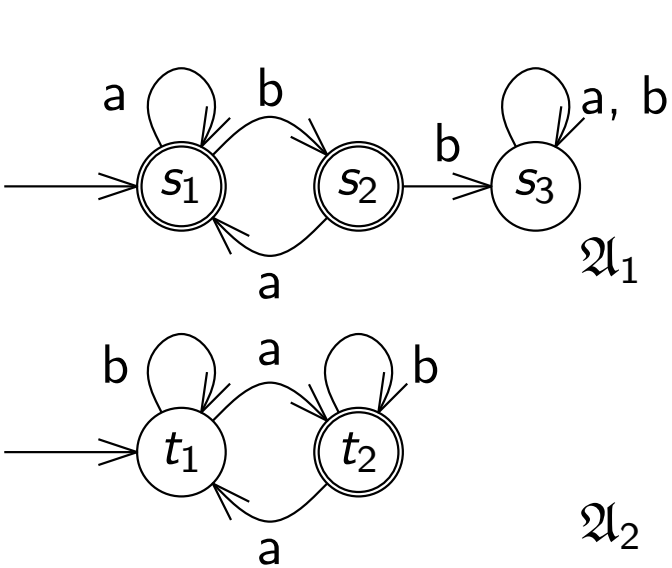


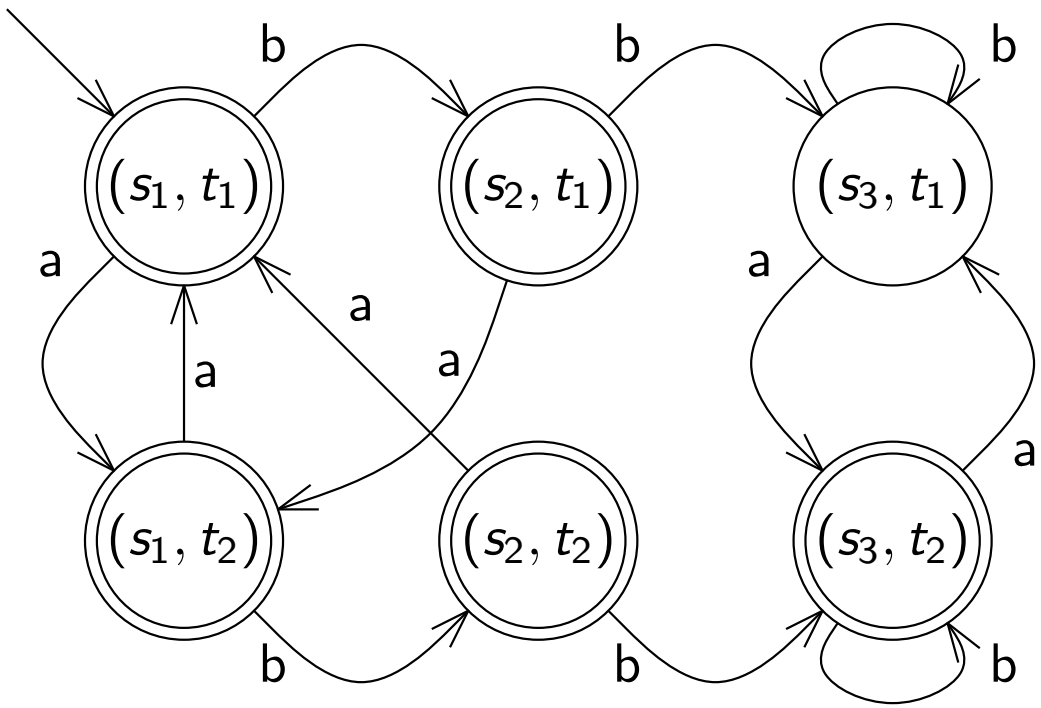
正則言語の性質 3 (証明は省略)

正則言語 $L_1, L_2 \subseteq \Sigma^*$ に対して，$L_1 \cup L_2 \subseteq \Sigma^*$ も正則言語

証明の雰囲気：

- $L_1 \cup L_2 = \Sigma^* - ((\Sigma^* - L_1) \cap (\Sigma^* - L_2))$ を使う





アルファベット Σ_1, Σ_2，言語 $L \subseteq (\Sigma_1 \times \Sigma_2)^*$

言語の射影 (projection) とは？

L の Σ_1 への射影とは，

$$\{a_1 a_2 \dots a_n \in \Sigma_1^* \mid \exists\ b_1 b_2 \dots b_n \in \Sigma_2^*\ ((a_1, b_1)(a_2, b_2) \dots (a_n, b_n) \in L)\}$$

例：$\Sigma_1 = \{\mathrm{a}, \mathrm{b}\}$, $\Sigma_2 = \{\mathrm{c}, \mathrm{d}\}$ のとき

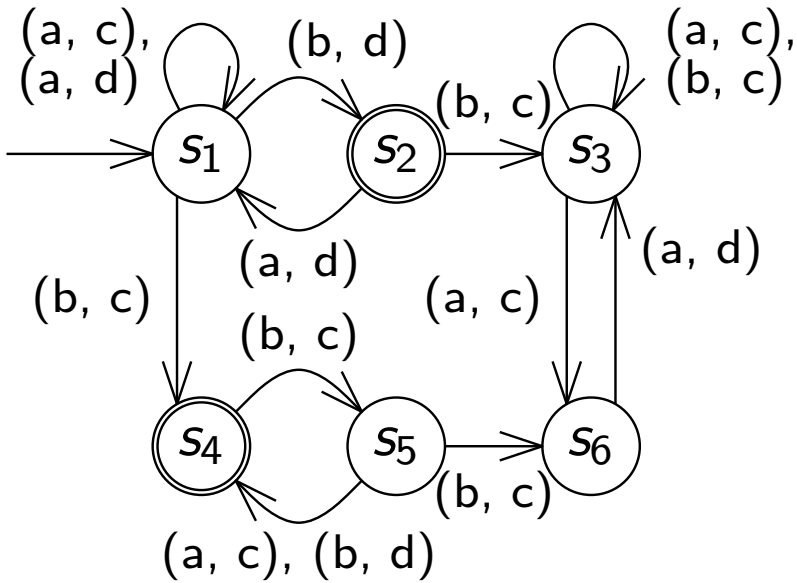
$$(\mathrm{a}, \mathrm{c})(\mathrm{a}, \mathrm{d})(\mathrm{b}, \mathrm{d}) \in L$$

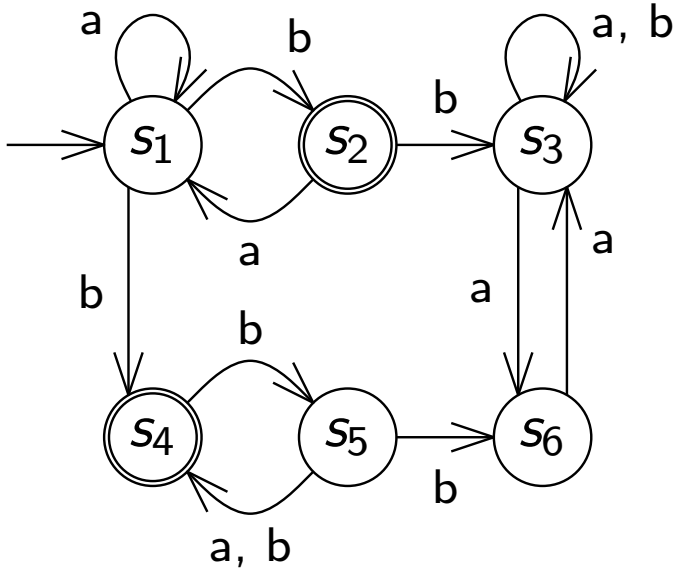
とすると，aab は L の Σ_1 への射影の要素

正則言語の性質 4 (証明は省略)

正則言語 $L \subseteq (\Sigma_1 \times \Sigma_2)^*$ の射影 $L' \subseteq \Sigma_1^*$ も正則言語

証明の雰囲気：





「非決定性」有限オートマトンを考えていることに注意

目次

Büchi の定理はオートマトンと単項二階論理をつなぐ

Büchi の定理

アルファベット Σ 上の言語 $L \subseteq \Sigma^*$ に対して

L が正則 $\Leftrightarrow$ ある単項二階論理式 ϕ が存在して，ϕ が L を定義する

証明の方針：

- 「$\Rightarrow$」L を受理するオートマトン $\mathfrak{A}$ から ϕ を作る
- 「$\Leftarrow$」ϕ から L を受理するオートマトン $\mathfrak{A}$ を作る

前回「$\Rightarrow$」の証明を行ったので，今回は「$\Leftarrow$」の証明を行う

Büchi の定理はオートマトンと単項二階論理をつなぐ

Büchi の定理

アルファベット Σ 上の言語 $L \subseteq \Sigma^*$ に対して

L が正則　$\Leftrightarrow$　ある単項二階論理式 ϕ が存在して，ϕ が L を定義する

「⇐」の証明の方針：

- ϕ から L を受理するオートマトン $\mathfrak{A}$ を作る

そのために以下を行う

1. ϕ を同値な形 φ に書き直す
2. 自由変数を持つ論理式も考慮しながら，オートマトンを構成する

論理式を書き直す方針

位置を表す変数「i」の使用を避ける

そのために，述語をいくつか用意する

$$
\begin{aligned}
\mathtt{singl}(X) &= X \text{ の要素数は } 1 \text{ である} \\
\mathtt{le}(X, Y) &= X \text{ の任意の要素は } Y \text{ の任意の要素よりも小さい} \\
\mathtt{symb}_{\mathtt{a}}(X) &= X \text{ の任意の要素に対して，その位置の文字は } \mathtt{a} \text{ である} \\
\mathtt{sub}(X, Y) &= X \text{ は } Y \text{ の部分集合である}
\end{aligned}
$$

まず，これらが単項二階論理式として書けることを確認する

論理式を書き直す方針

位置を表す変数「i」の使用を避ける

実際に，以下の通り，単項二階論理式として書ける

$$
\begin{aligned}
\mathtt{singl}(X) &= \exists\, i\, (i \in X \wedge \forall\, j\, (j \in X \to i = j)) \\
\mathtt{le}(X, Y) &= \forall\, i\, (\forall\, j\, ((i \in X \wedge j \in Y) \to i \leq j)) \\
\mathtt{symb}_{\mathsf{a}}(X) &= \forall\, i\, (i \in X \to w_i = \mathsf{a}) \\
\mathtt{sub}(X, Y) &= \forall\, i\, (i \in X \to i \in Y)
\end{aligned}
$$

用意した述語を使って，位置を表す変数の使用を避ける

そのために，変数 i を集合 $I = \{i\}$ に置き換える，と考える

$\therefore$ 単項二階論理式に現れる述語や量化も次のように置き換えられる

$$
\begin{aligned}
\forall\, i\, (\phi) &\rightsquigarrow \forall\, I\, (\mathtt{singl}(I) \to \phi) \\
\exists\, i\, (\phi) &\rightsquigarrow \exists\, I\, (\mathtt{singl}(I) \wedge \phi) \\
i = j &\rightsquigarrow \mathtt{le}(I, J) \wedge \mathtt{le}(J, I) \\
i \in X &\rightsquigarrow \mathtt{sub}(I, X) \\
w_i = \mathsf{a} &\rightsquigarrow \mathtt{symb}_{\mathsf{a}}(I) \\
i \leq j &\rightsquigarrow \mathtt{le}(I, J)
\end{aligned}
$$

$\therefore$ 単項二階論理式の変数は「位置の集合」を表すもののみと仮定できる

証明：論理式の書き直し (4)

書き換えられた論理式は次のものから構成される

変数

- 位置の部分集合 $X \subseteq [n]$

論理結合子

- $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$

量化子

- $\forall, \exists$

定数

- Σ の要素

述語

- $X \subseteq [n]$ に対して，$\mathtt{singl}(X)$
- $X, Y \subseteq [n]$ に対して，$\mathtt{le}(X, Y)$
- $X \subseteq [n], \mathsf{a} \in \Sigma$ に対して，$\mathtt{symb}_{\mathsf{a}}(X)$
- $X, Y \subseteq [n]$ に対して，$\mathtt{sub}(X, Y)$

ただし，$[n] = \{1, 2, \ldots, n\}$

Büchi の定理はオートマトンと単項二階論理をつなぐ

Büchi の定理

アルファベット Σ 上の言語 $L \subseteq \Sigma^*$ に対して

L が正則 $\Leftrightarrow$ ある単項二階論理式 ϕ が存在して，ϕ が L を定義する

「⇐」の証明の方針：

- ϕ から L を受理するオートマトン $\mathfrak{A}$ を作る

そのために以下を行う

1. ϕ を同値な形 φ に書き直す (終了)
2. 自由変数を持つ論理式も考慮しながら，オートマトンを構成する

自由変数を持つ単項二階論理式

- いま，単項二階論理式に現れる変数は「位置の集合」を表すもののみ
- φ の自由変数を $X_1, X_2, \ldots, X_k$ とする

$$\varphi(X_1, X_2, \ldots, X_k)$$

言語 $L \subseteq \Sigma^*$ が次を満たすとする

- $w = w_1 \ldots w_n \in L \Leftrightarrow$ ある $P_1, P_2, \ldots, P_k \subseteq [n]$ が存在して

$$w \models \varphi(P_1, P_2, \ldots, P_k)$$

目標

φ から L を受理するオートマトンを作る

証明：オートマトンの構成 (1)

単項二階論理式 $\varphi(X_1, X_2, \ldots, X_k)$

基本的な考え方

- アルファベットとして，$(\Sigma \times \{0,1\}^k)^*$ を考える
- 最終的に，$(\Sigma \times \{0,1\}^k)^*$ から Σ^* へ射影する

- つまり，$w \models \varphi(P_1, P_2, \ldots, P_k)$ を満たす $w = w_1 w_2 \ldots w_n \in \Sigma^*$ に対して，

  $$w_i \in \Sigma \text{ を } (w_i, b_{i1}, b_{i2}, \ldots, b_{ik}) \in \Sigma \times \{0,1\}^k \text{ に変換する}$$

- 変換規則：$b_{ij} = 1 \Leftrightarrow i \in P_j$

例

$\varphi(X_1) = \exists\ Y\ (\mathtt{singl}(Y) \wedge \mathtt{symb}_{\mathsf{a}}(Y) \wedge \mathtt{sub}(Y, X_1))$

- $w = \mathsf{abaa}$ とすると，

$$w \models \varphi(\{1, 2\})$$

- $\because X_1 = \{1, 2\}, Y = \{1\} \Rightarrow \mathtt{singl}(Y) \wedge \mathtt{symb}_{\mathsf{a}}(Y) \wedge \mathtt{sub}(Y, X_1)$ は真
- このとき，$w = \mathsf{abaa}$ は次の文字列に変換される

$$(\mathsf{a}, 1)(\mathsf{b}, 1)(\mathsf{a}, 0)(\mathsf{a}, 0)$$

- また，他の文字列にも変換される

一般に，$L = \{w = w_1 \dots w_n \mid \exists\ P_1, \dots, P_k\ (w \models \varphi(P_1, \dots, P_k))\}$ のとき，

$$L^* = \left\{ (w_1, b_{11}, \dots, b_{1k}) \dots (w_n, b_{n1}, \dots, b_{nk}) \;\middle|\; \begin{array}{l} w \models \varphi(P_1, \dots, P_k), \\ b_{ij} = 1 \text{ iff } i \in P_j \end{array} \right\}$$

とする

単項二階論理式 $\varphi(X_1, X_2, \ldots, X_k)$

基本的な考え方 (2)

$\varphi(X_1, X_2, \ldots, X_k)$ の構造に従って再帰的にオートマトンを構成する

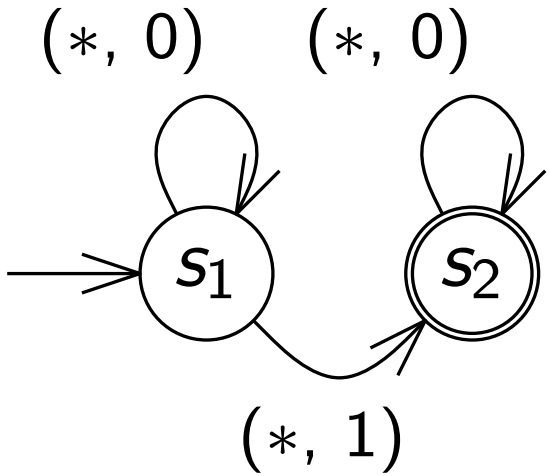
まずは，述語に対応するオートマトンを構成する

- $\varphi_1(X_1) = \mathtt{singl}(X_1)$
- $\varphi_2(X_1, X_2) = \mathtt{le}(X_1, X_2)$
- $\varphi_3(X_1) = \mathtt{symb}_{\mathtt{a}}(X_1)$
- $\varphi_4(X_1, X_2) = \mathtt{sub}(X_1, X_2)$

証明：オートマトンの構成 (3) — singl

$\varphi_1(X_1) = \mathtt{singl}(X_1)$

- 考えるアルファベットは $\Sigma \times \{0, 1\}$

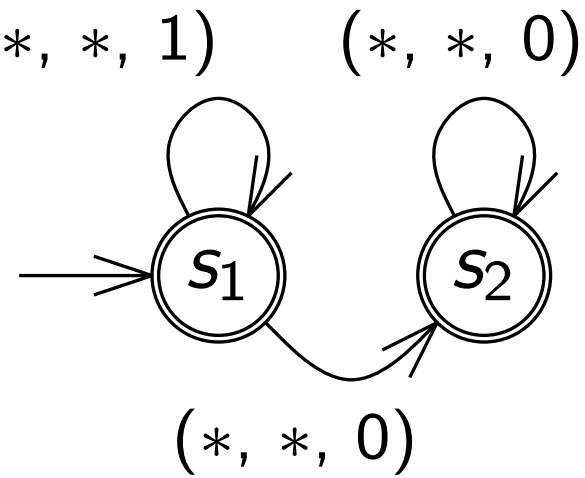


「$*$」はワイルドカード (Σ の任意の文字を表す)

証明：オートマトンの構成 (3) — le

$\varphi_2(X_1, X_2) = \mathtt{le}(X_1, X_2)$

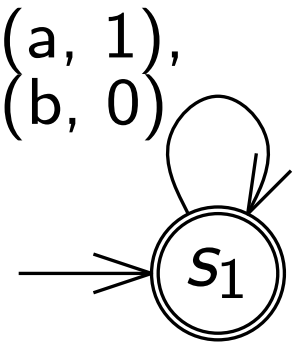
- 考えるアルファベットは $\Sigma \times \{0,1\}^2$



証明：オートマトンの構成 (3) — symb

$\varphi_3(X_1) = \mathtt{symb}_{\mathsf{a}}(X_1)$

- 考えるアルファベットは $\Sigma \times \{0, 1\}$

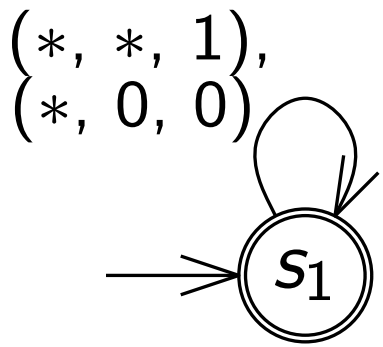


これは $\Sigma = \{\mathsf{a}, \mathsf{b}\}$ のとき

証明：オートマトンの構成 (4) — sub

$\varphi_4(X_1, X_2) = \mathrm{sub}(X_1, X_2)$

- 考えるアルファベットは $\Sigma \times \{0,1\}^2$



単項二階論理式 $\varphi(X_1, X_2, \ldots, X_k)$

基本的な考え方 (2) 再掲

$\varphi(X_1, X_2, \ldots, X_k)$ の構造に従って再帰的にオートマトンを構成する

次は，結合子，量化に対応するオートマトンを再帰的に構成する

- $\varphi = \neg\varphi_1$
- $\varphi = \varphi_1 \wedge \varphi_2$
- $\varphi = \varphi_1 \vee \varphi_2$
- $\varphi = \varphi_1 \rightarrow \varphi_2$
- $\varphi = \varphi_1 \leftrightarrow \varphi_2$
- $\varphi = \exists\, X\ \varphi_1(X)$
- $\varphi = \forall\, X\ \varphi_1(X)$

単項二階論理式 $\varphi(X_1, X_2, \ldots, X_k)$

基本的な考え方 (2) 再掲

$\varphi(X_1, X_2, \ldots, X_k)$ の構造に従って再帰的にオートマトンを構成する

次は，結合子，量化に対応するオートマトンを再帰的に構成する

- $\varphi = \neg\varphi_1$
- $\varphi = \varphi_1 \wedge \varphi_2$
- $\varphi = \varphi_1 \vee \varphi_2$
- $\varphi = \varphi_1 \to \varphi_2$
- $\varphi = \varphi_1 \leftrightarrow \varphi_2$
- $\varphi = \exists\, X\ \varphi_1(X)$
- $\varphi = \forall\, X\ \varphi_1(X)$

注：$\varphi_1 \vee \varphi_2 = \neg(\neg\varphi_1 \wedge \neg\varphi_2)$, $\varphi_1 \to \varphi_2 = \neg\varphi_1 \vee \varphi_2$,
$\varphi_1 \leftrightarrow \varphi_2 = (\varphi_1 \to \varphi_2) \wedge (\varphi_2 \to \varphi_1)$, $\forall\, X\ \varphi_1(X) = \neg\exists\, X\ (\neg\varphi_1)$

証明：オートマトンの構成 (5) — $\neg$, $\wedge$

$\varphi = \neg\varphi_1$

- φ_1 からオートマトン $\mathfrak{A}_1$ が構成できていると仮定
- $\mathfrak{A}_1$ が受理する言語が L_1 であるとする
- このとき，$\overline{L_1}$ を受理するオートマトン $\mathfrak{A}$ が存在
- 「$w \in \overline{L_1} \Leftrightarrow w \models \neg\varphi_1$」なので，
  $\mathfrak{A}$ が欲しかったオートマトン

$\varphi = \varphi_1 \wedge \varphi_2$

- φ_1, φ_2 からオートマトン $\mathfrak{A}_1, \mathfrak{A}_2$ が構成できていると仮定
- $\mathfrak{A}_1, \mathfrak{A}_2$ が受理する言語が L_1, L_2 であるとする
- このとき，$L_1 \cap L_2$ を受理するオートマトン $\mathfrak{A}$ が存在
- 「$w \in L_1 \cap L_2 \Leftrightarrow w \models \varphi_1 \wedge \varphi_2$」なので，
  $\mathfrak{A}$ が欲しかったオートマトン

証明：オートマトンの構成 (5) — $\exists$

$\varphi = \exists\, X\ (\varphi_1(X))$

- $\varphi_1(X)$ からオートマトン $\mathfrak{A}_1$ が構成できていると仮定
- $\mathfrak{A}_1$ が受理する言語が L_1 であるとする
- $w \in L_1 \Leftrightarrow$ ある P が存在して，$w \models \varphi_1(P)$
- つまり，L_1 を X に対応する成分以外に射影した言語を L とすると，$w \in L \Leftrightarrow w \models \exists\, X\ (\varphi_1(X))$
- したがって，L を受理するオートマトンが欲しかったオートマトン

単項二階論理式 $\varphi(X_1, X_2, \ldots, X_k)$

基本的な考え方

- アルファベットとして，$(\Sigma \times \{0,1\}^k)^*$ を考える
- 最終的に，$(\Sigma \times \{0,1\}^k)^*$ から Σ^* へ射影する

基本的な考え方 (2)

$\varphi(X_1, X_2, \ldots, X_k)$ の構造に従って再帰的にオートマトンを構成する

- 構成されたオートマトンが受理する言語は $(\Sigma \times \{0,1\}^k)^*$ の部分集合
- これを Σ^* は射影すると，$\varphi(X_1, \ldots, X_k)$ が定義する言語が得られる
- つまり，ϕ の定義する言語は正則 □

Büchi の定理はオートマトンと単項二階論理をつなぐ

Büchi の定理

アルファベット Σ 上の言語 $L \subseteq \Sigma^*$ に対して

L が正則 $\Leftrightarrow$ ある単項二階論理式 ϕ が存在して，ϕ が L を定義する

Büchi の定理の「$\Leftarrow$」の証明の帰結

与えられた単項二階論理式 ϕ に対して，
文字列 w が ϕ の定義する言語 L の要素かどうか判定するアルゴリズムで
$O(f(|\phi|) + |w|)$ 時間で動くものが存在する

- $f(\cdot)$ はある関数，$|\phi|$ は ϕ の長さ，$|w|$ は w の長さ
- これが Courcelle の定理の証明に効いてくる

目次

無向グラフ $G = (V, E)$

グラフに対する単項二階論理式

変数

- 頂点 $v \in V$
- 辺 $e \in E$
- 頂点部分集合 $X \subseteq V$
- 辺部分集合 $F \subseteq E$

論理結合子

- $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$

量化子

- $\forall$, $\exists$

述語

- $u, v \in V$ に対して,「$u = v$」
- $e, f \in E$ に対して,「$e = f$」
- $v \in V$, $X \subseteq V$ に対して,「$v \in X$」
- $e \in E$, $F \subseteq E$ に対して,「$e \in F$」
- $X_1, X_2 \subseteq V$ に対して,「$X_1 \subseteq X_2$」
- $F_1, F_2 \subseteq E$ に対して,「$F_1 \subseteq F_2$」
- $u, v \in V$ に対して,「$\mathtt{adj}(u, v)$」
- $v \in V$, $e \in E$ に対して,「$\mathtt{inc}(v, e)$」

これらを用いて記述できる論理式が単項二階論理式

Courcelle の定理 (1990)

無向グラフ G と単項二階論理式 ϕ が与えられたとき，
$G \models \phi$ となるか判定することは $O(f(\mathtt{tw}(G), |\phi|)|V|)$ 時間でできる

- ここで，$|\phi|$ は ϕ の長さ (記述長) で，f はある関数
- $\mathtt{tw}(G)$ と $|\phi|$ が定数ならば，この計算量は $O(|V|)$

Courcelle の定理の帰結

G の木幅が定数であれば，単項二階論理式で表現できる性質の判定は
線形時間でできる

重要な点

- 判定したい性質を単項二階論理式で書くだけで，自動的に，
線形時間アルゴリズムが得られる
- 様々な性質に対するアルゴリズムを包括した「メタアルゴリズム」

Courcelle の定理：証明のアイディア

1. グラフに対する単項二階論理式を
   木に対する単項二階論理式に変換する
2. オートマトンではなく，木オートマトンを考える
   - Büchi の定理に似た定理を木オートマトンで証明する
     (Doner '70, Thatcher, Wright '68)
3. 単項二階論理式が木分解によって満たされるか，判定すればよい

木オートマトン (直感)

文字列ではなく，木 (順序付き二分木) を受理する

木オートマトンとは？

次の 4 つから構成される $\mathfrak{A}$

- 有限集合 S : 状態の集合
- 有限集合 Σ : アルファベット
- $\Delta \subseteq (S \cup \{\bot\}) \times (S \cup \{\bot\}) \times \Sigma \times S$: 遷移関係
- $F \subseteq S$: 受理状態の集合

$\mathfrak{A} = (S, \Sigma, \Delta, F)$ と書くこともある

これは，より正確にいえば
「順序付き二分木に対するボトムアップ非決定性木オートマトン」の定義

順序付き二分木 $T = (V, E)$，頂点ラベル $\lambda\colon V \to \Sigma$

木オートマトンによるラベル付き木の受理

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ がラベル付き木 (T, λ) を受理するとは，次を満たす写像 $\rho\colon V \to S$ が存在すること

- 節点 $v \in V$ が子を 2 つ持ち，それらが (順に) v_1, v_2 であるとき，$(\rho(v_1), \rho(v_2), \lambda(v), \rho(v)) \in \Delta$
- 節点 $v \in V$ が左の子のみを持ち，それが v_1 であるとき，$(\rho(v_1), \bot, \lambda(v), \rho(v)) \in \Delta$
- 節点 $v \in V$ が右の子のみを持ち，それが v_2 であるとき，$(\bot, \rho(v_2), \lambda(v), \rho(v)) \in \Delta$
- 節点 $v \in V$ が葉であるとき (子を持たないとき)，$(\bot, \bot, \lambda(v), \rho(v)) \in \Delta$
- 節点 $v \in V$ が根であるとき，$\rho(v) \in F$

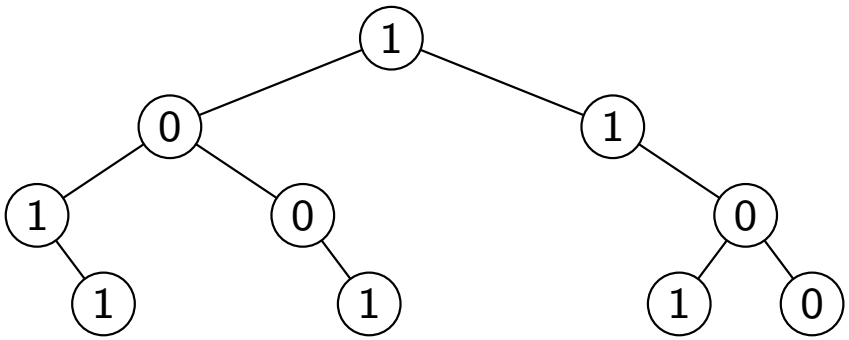
雰囲気：葉から根に向かって状態遷移を行い，根が受理状態となる

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる

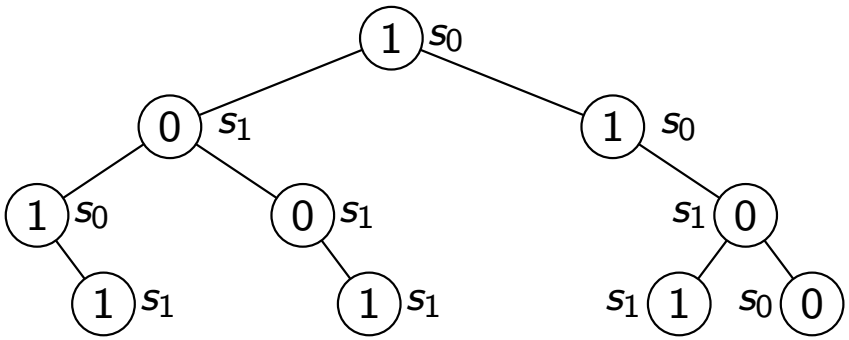


木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる

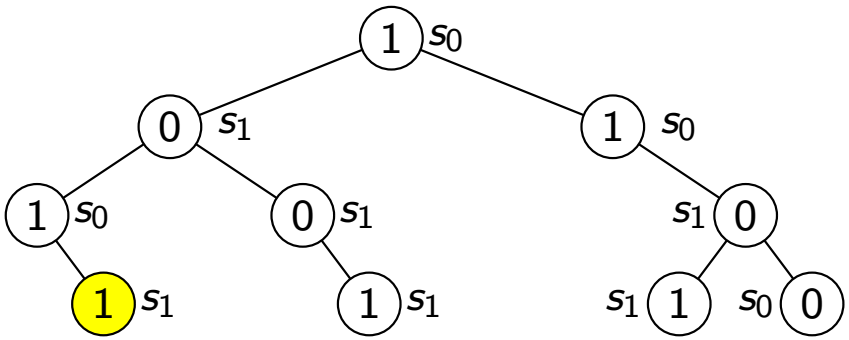


木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



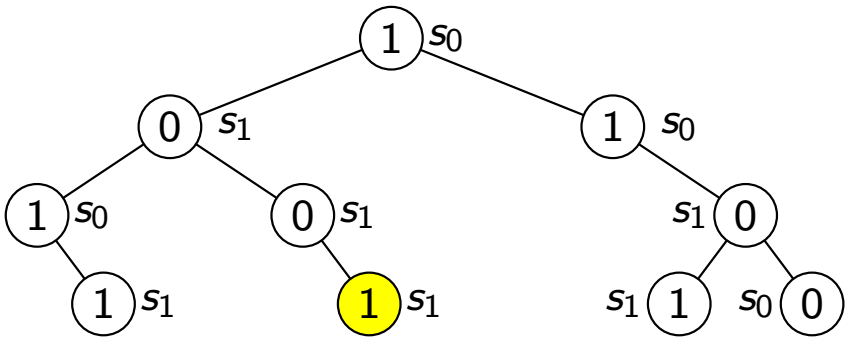
$(\bot, \bot, 1, s_1) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



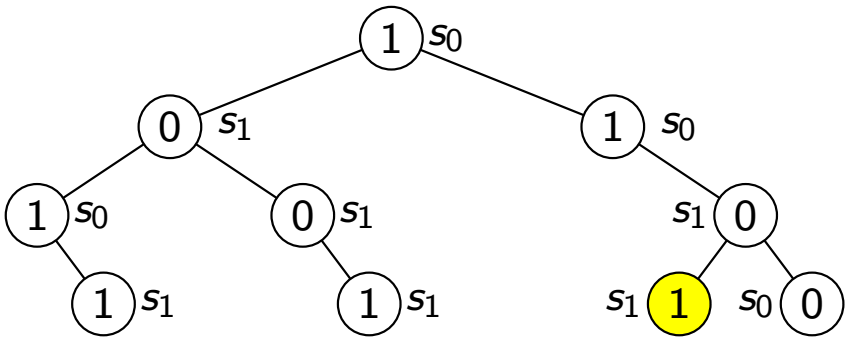
$(\bot, \bot, 1, s_1) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



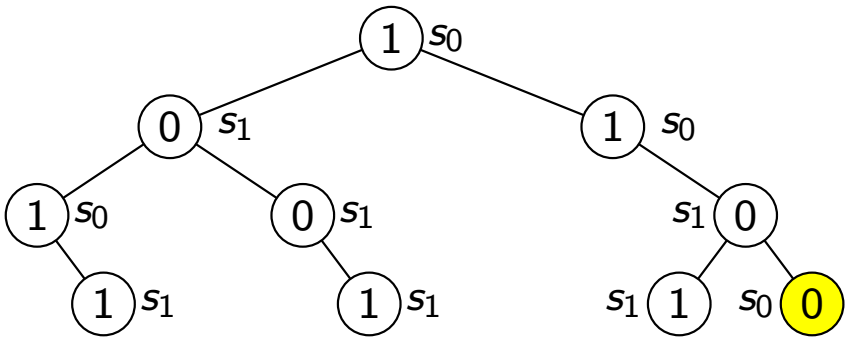
$(\bot, \bot, 1, s_1) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



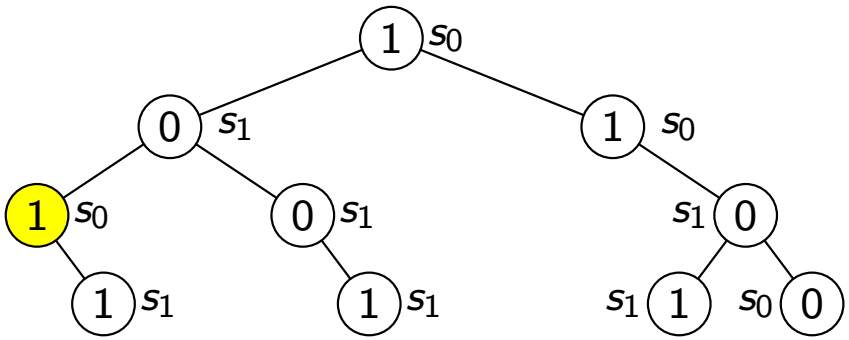
$(\bot, \bot, 0, s_0) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



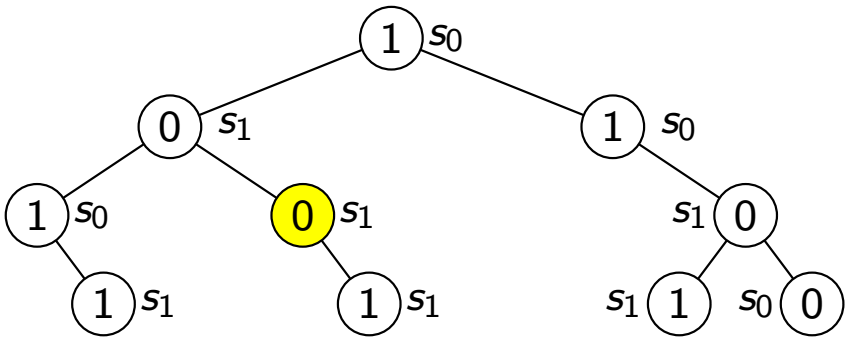
$(\bot, s_1, 1, s_0) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



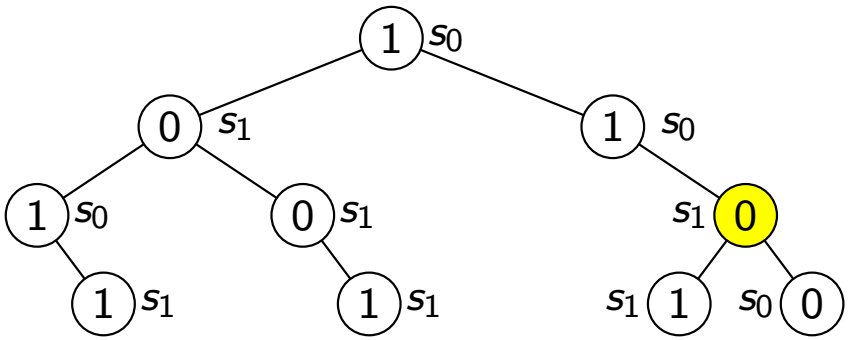
$(\bot, s_1, 0, s_1) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



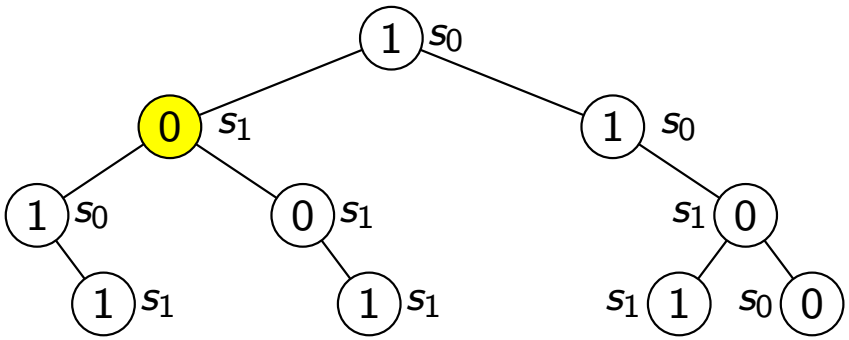
$(s_1, s_0, 0, s_1) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



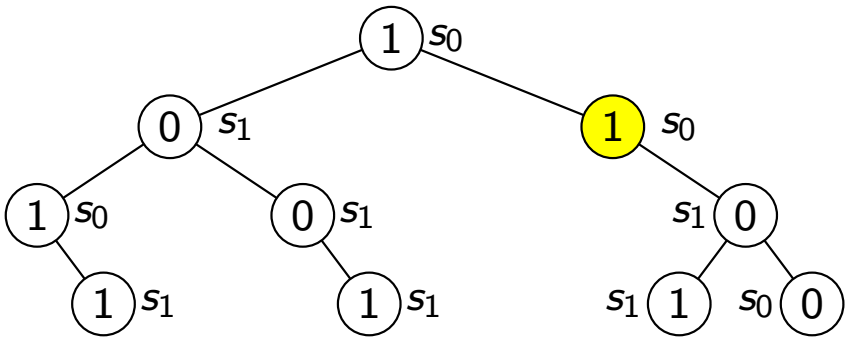
$(s_0, s_1, 0, s_1) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



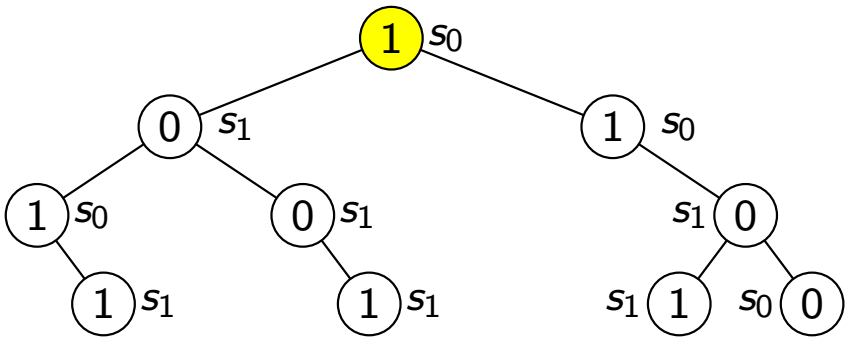
$(\bot, s_1, 1, s_0) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



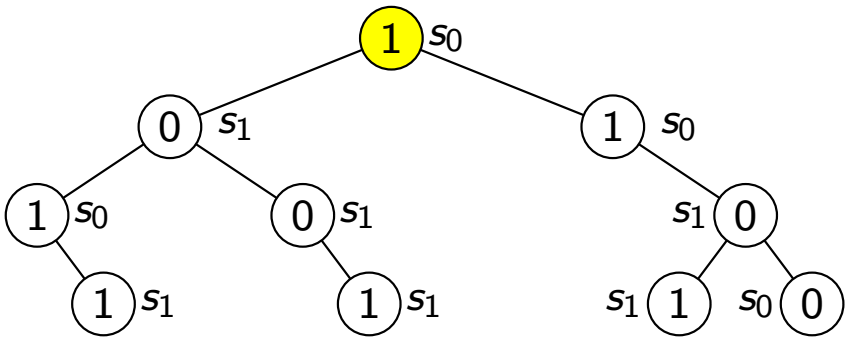
$(s_1, s_0, 1, s_0) \in \Delta$

木オートマトン：例

木オートマトン $\mathfrak{A} = (S, \Sigma, \Delta, F)$ を次のように定義

- $S = \{s_0, s_1\}$, $\Sigma = \{0, 1\}$, $F = \{s_0\}$
- $\Delta = \{(\bot, \bot, 0, s_0), (\bot, \bot, 1, s_1)\} \cup$
  $\{(s_i, \bot, j, s_k), (\bot, s_i, j, s_k) \mid i, j, k \in \{0, 1\}, i + j \equiv k \pmod 2\} \cup$
  $\{(s_i, s_j, k, s_\ell) \mid i, j, k, \ell \in \{0, 1\}, i + j + k \equiv \ell \pmod 2\}$

$\mathfrak{A}$ が次のラベル付き木を受理する様子を見てみる



$s_0 \in F$

目次

今日のまとめと次回の予告

今日の目標

- Büchi の定理の証明の残り (アルゴリズム)
- Courcelle の定理の証明の雰囲気

次回と次々回

幅の小さな木分解をどのように作るのか？ (アルゴリズム)

残った時間の使い方

- 演習問題をやる
  - 相談推奨 (ひとりでやらない)
- 質問をする
  - 教員は巡回
- 退室時，小さな紙に感想など書いて提出する ← 重要
  - 内容は何でも OK
  - 匿名で OK

目次