ファイブレーションを用いた分割細分アルゴリズムと Hopcroft の最適化の構造

眞田 嵩大 [1]

[1] 京都大学 数理解析研究所

CSCAT 2023

joint work with 小島 良太, 小森田 祐一, 室屋 晃子, 蓮尾 一郎

目次

DFA に対する Hopcroft の分割細分アルゴリズム

重み付き木に対する Hopcroft の不等式

ファイブレーションを用いた分割細分アルゴリズム

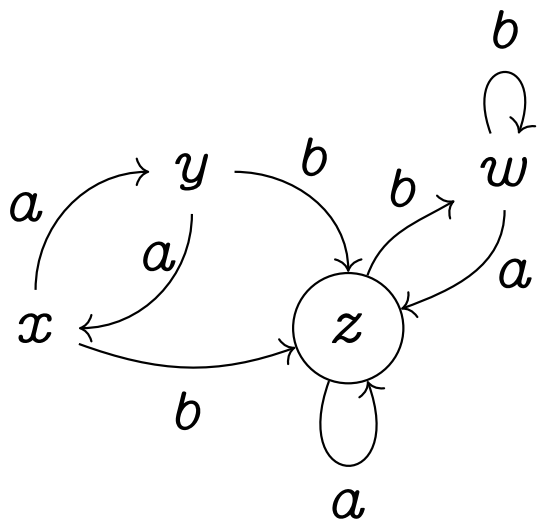
状態の振る舞い

文字集合 Σ を固定する。
決定性有限オートマトン (DFA) $\mathcal{A} = (C, \delta, C_{\mathrm{acc}})$ があるとする。

- 状態集合 C
- 遷移関数 $\delta : C \times \Sigma \to C$
- 受理状態 $C_{\mathrm{acc}} \subseteq C$

ふたつの状態 $x, y \in C$ が同じ振る舞いをするかどうかを判定したい。

例：$\Sigma = \{a, b\}$, $C = \{x, y, z, w\}$, $F = \{z\}$



双模倣性関係

DFA $\mathcal{A} = (C, \delta, C_{\mathrm{acc}})$。
関係 $R \subseteq C \times C$ が双模倣であるとは、任意の $(x, y) \in R$ について

- $x \in C_{\mathrm{acc}} \iff y \in C_{\mathrm{acc}}$ かつ
- 任意の $a \in \Sigma$ について $(\delta(x, a), \delta(y, a)) \in R$ であること。

$$\begin{array}{ccc} x & \xrightarrow{a} & x' \\ {\scriptstyle R}\,\big| & & \big|\,{\scriptstyle R} \\ y & \xrightarrow[a]{} & y' \end{array}$$

- 双模倣は $\cup$ で閉じる: 双模倣の族 $\{R_i\}_{i \in I}$ について $\bigcup_{i \in I} R_i$ は双模倣。
- 最大の双模倣関係を $(\sim) \subseteq C \times C$ で表し双模倣性関係と呼ぶ。
- $\sim$ は同値関係になる。

$x \sim y$ のとき、x と y は同じ振る舞いをすると考える。

最大不動点としての双模倣性関係

C 上の同値関係とその間の包含がなす圏を $\mathbf{EqRel}_C$ と書く。

$$\Phi\colon \mathbf{EqRel}_C \to \mathbf{EqRel}_C$$

$$\Phi_{\mathcal{A}}(R) := \left\{ (x,y) \in C \times C \;\middle|\; \begin{array}{l} x \in C_{\mathrm{acc}} \Leftrightarrow y \in C_{\mathrm{acc}} \quad \text{かつ} \\ \forall a \in \Sigma.\ \begin{array}{r} x \xrightarrow{a} \delta(x,a) \\ \mid R \\ y \xrightarrow[a]{} \delta(y,a) \end{array} \end{array} \right\}$$

双模倣性関係 $(\sim) \in \mathbf{EqRel}_C$ は $\Phi_{\mathcal{A}}$ の最大不動点 $\nu(\Phi_{\mathcal{A}})$ である。

$$\top \supseteq \Phi(\top) \supseteq \Phi^2(\top) \supseteq \cdots \supseteq \nu(\Phi) = \bigcap_{n \in \omega} \Phi^n(\top) = (\sim)$$

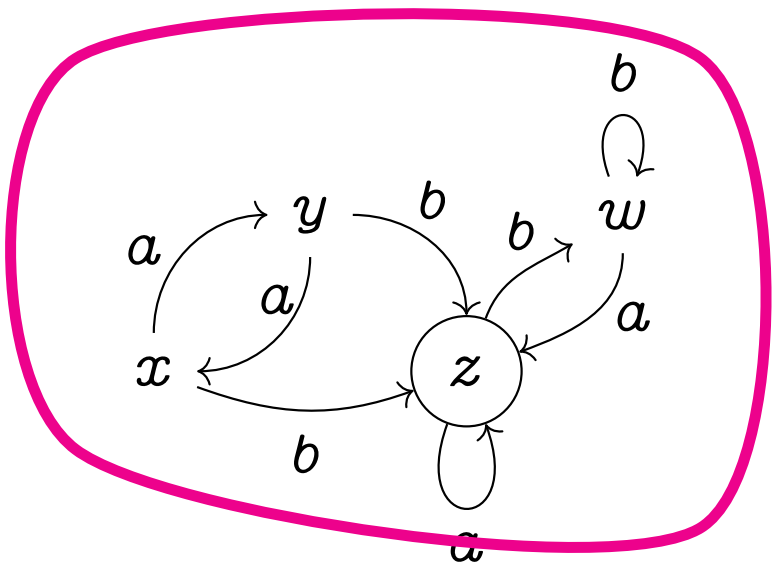
分割細分アルゴリズム

$\top$ からはじめて、同値類をどんどん分割していくことにより $\sim$ を得る。

$$\Phi_{\mathcal{A}}(R) := \left\{ (x, y) \in C \times C \;\middle|\; \begin{array}{l} x \in C_{\mathrm{acc}} \Leftrightarrow y \in C_{\mathrm{acc}} \quad \text{かつ} \\ \forall a \in \Sigma.\, (\delta(x, a), \delta(y, a)) \in R \end{array} \right\}$$

$$\top \supseteq \Phi(\top) \supseteq \Phi^2(\top) \supseteq \cdots \supseteq \nu(\Phi) = \bigcap_{n \in \omega} \Phi^n(\top) = (\sim)$$

$\top =$



すべての状態は同じ振る舞いをするに違いない！

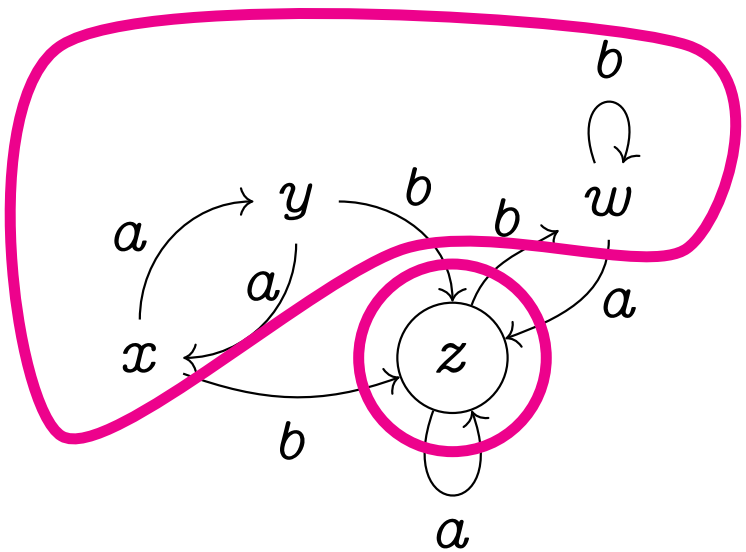
分割細分アルゴリズム

$\top$ からはじめて、同値類をどんどん分割していくことにより $\sim$ を得る。

$$\Phi_{\mathcal{A}}(R) := \left\{ (x, y) \in C \times C \;\middle|\; \begin{array}{l} x \in C_{\mathrm{acc}} \Leftrightarrow y \in C_{\mathrm{acc}} \quad \text{かつ} \\ \forall a \in \Sigma.\, (\delta(x, a), \delta(y, a)) \in R \end{array} \right\}$$

$$\top \supseteq \Phi(\top) \supseteq \Phi^2(\top) \supseteq \cdots \supseteq \nu(\Phi) = \bigcap_{n \in \omega} \Phi^n(\top) = (\sim)$$

$\Phi(\top) =$



いや、さすがに受理状態かどうかは
気にしないといけなかった…

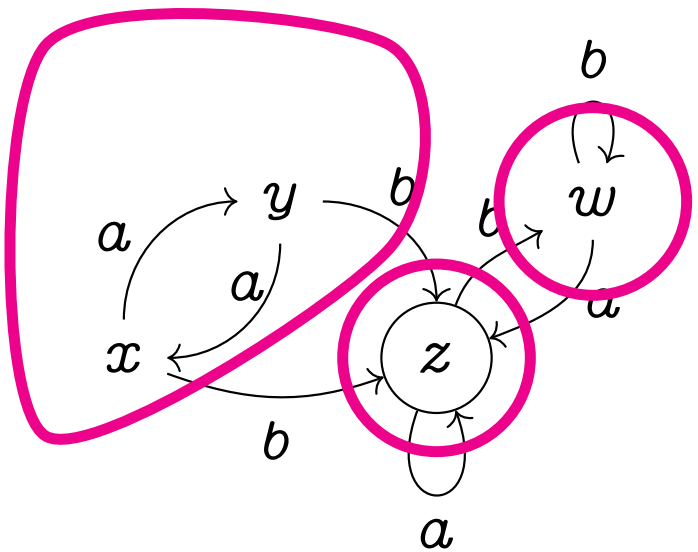
分割細分アルゴリズム

$\top$ からはじめて、同値類をどんどん分割していくことにより $\sim$ を得る。

$$\Phi_{\mathcal{A}}(R) := \left\{ (x, y) \in C \times C \;\middle|\; \begin{array}{l} x \in C_{\mathrm{acc}} \Leftrightarrow y \in C_{\mathrm{acc}} \quad \text{かつ} \\ \forall a \in \Sigma.\, (\delta(x, a), \delta(y, a)) \in R \end{array} \right\}$$

$$\top \supseteq \Phi(\top) \supseteq \Phi^2(\top) \supseteq \cdots \supseteq \nu(\Phi) = \bigcap_{n \in \omega} \Phi^n(\top) = (\sim)$$

$\Phi^2(\top) =$



各文字での遷移先が受理状態かどうかも気にしないといけなかった…

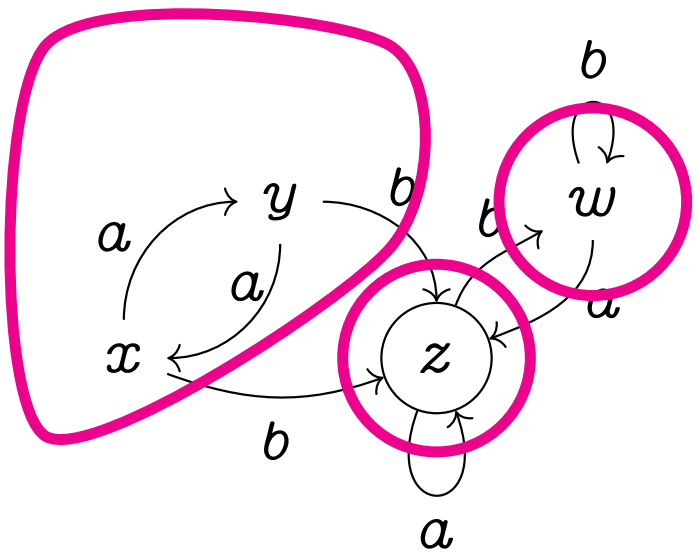
分割細分アルゴリズム

$\top$ からはじめて、同値類をどんどん分割していくことにより $\sim$ を得る。

$$\Phi_{\mathcal{A}}(R) := \left\{ (x,y) \in C \times C \;\middle|\; \begin{array}{l} x \in C_{\mathrm{acc}} \Leftrightarrow y \in C_{\mathrm{acc}} \quad \text{かつ} \\ \forall a \in \Sigma.\, (\delta(x,a), \delta(y,a)) \in R \end{array} \right\}$$

$$\top \supseteq \Phi(\top) \supseteq \Phi^2(\top) \supseteq \cdots \supseteq \nu(\Phi) = \bigcap_{n \in \omega} \Phi^n(\top) = (\sim)$$

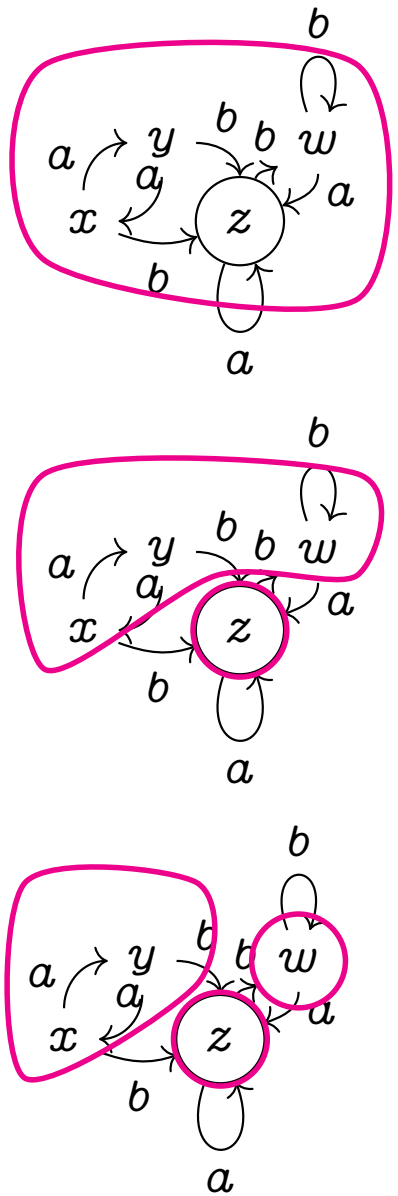
$\Phi^3(\top) =$

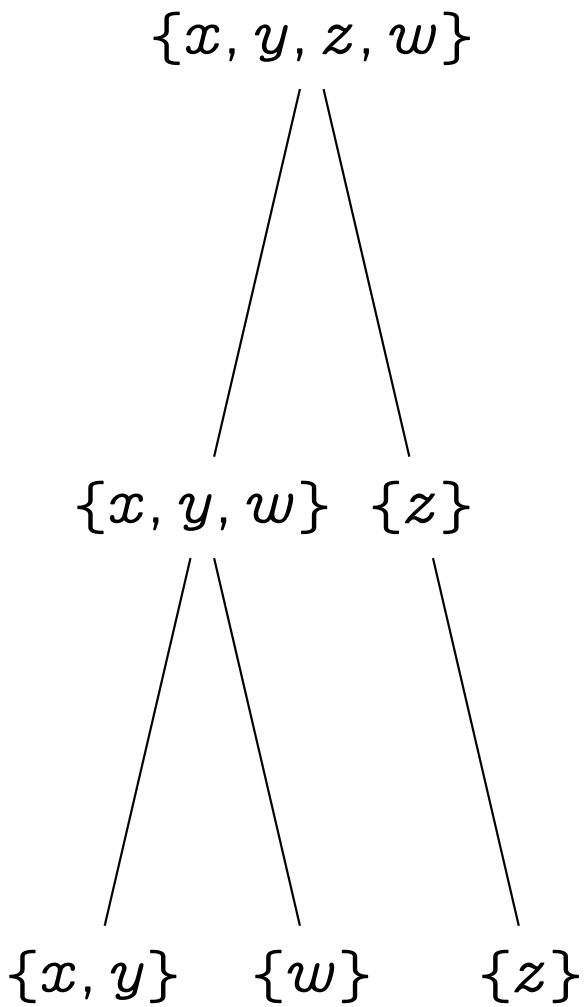


各文字での遷移先が $\Phi^2(\top)$ で区別されているかどうかも気にするべきだった…

だが、それを気にしたとしても以前の分割 $\Phi^2(\top)$ と変わらなかった！
つまり $\Phi^3(\top) = \Phi^2(\top)$ であり、
ゆえに $(\sim) = \nu(\Phi) = \Phi^3(\top)$。
これが求めたかった双模倣性関係（の同値類）だ！

細分のようすは木で表される





木のつくりかたを工夫することで、状態数 n に対して $\mathcal{O}(n \log n)$ で双模倣性関係 $\sim$ を計算できる [Hopcroft. 1971]

しかし、その計算量の議論はとても複雑だった

Hopcroft の論文を解読した論文：

- Gries. 1973
- Knuutila. 2001

やはり複雑。

我々の貢献 (1)：重み付き木に対する不等式を導出して議論の本質を定式化する

DFA に対する Hopcroft の分割細分アルゴリズム

重み付き木に対する Hopcroft の不等式

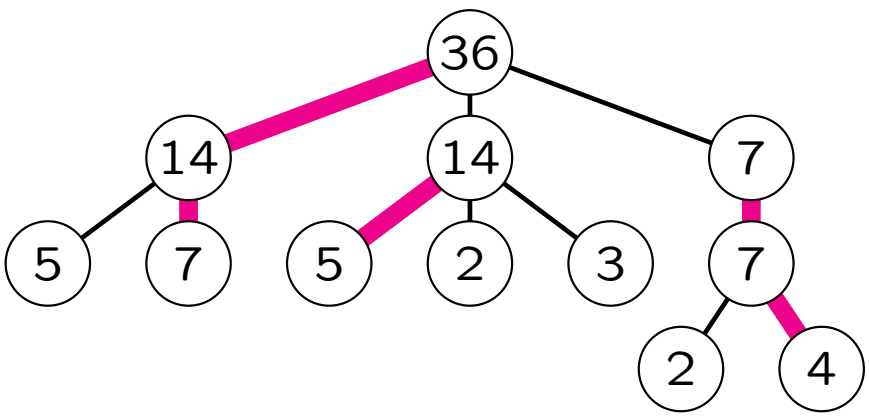
ファイブレーションを用いた分割細分アルゴリズム

重み付き木

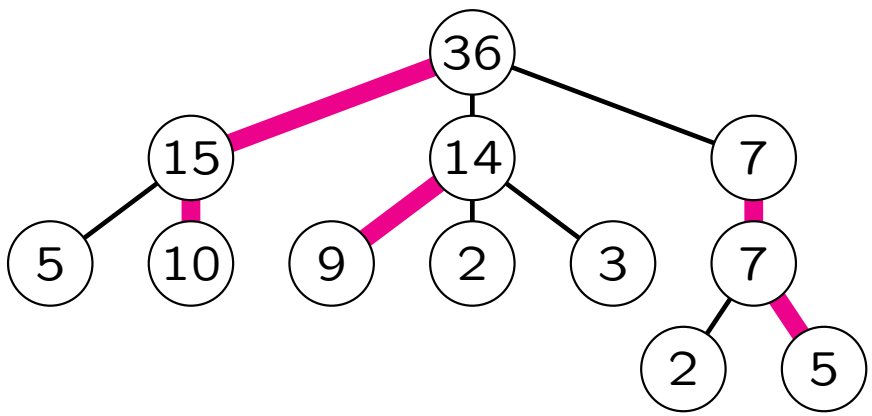
根付き木 T に対して、その重み関数 w とは、関数 $V(T) \to \mathbb{N}$ であって、どんな頂点 $v \in V(T)$ についても

$$\sum_{u \in \mathrm{ch}(v)} w(u) \leq w(v)$$

であるもののこと。等号 $=$ が成り立っているとき重み関数 w はタイトであるという。
木 T とその重み関数 w に対して重い子の指定 h は、葉ではない各頂点 v に対する最大の重みを持つ子 $h(v)$ の割り当てのこと。



タイトではない重み関数と重い子の指定



タイトな重み関数と重い子の指定

主定理 1（Hopcroft の不等式）

Theorem

根付き木 T と重み関数 w、重い子の指定 h に対して次の不等式が成り立つ。

$$\sum_{v \in V(T)} \sum_{\substack{u \in \mathrm{ch}(v) \\ u \neq h(v)}} w(u) \leq w(r) \log_2 w(r) + \sum_{T \text{ の葉 } l} w(l) \log_2 w(l)$$

ここで r は T の根である。

これを証明するために 3 つの観察を行う

1. Hopcroft のトリック（よく知られている）
2. 縦 × 横 = 横 × 縦
3. 重み関数のタイト化

Hopcroft のトリック

根付き木 T と重み関数 w、重い子の指定 h があるとする。
頂点 $v, v' \in V(T)$ に対して、次のように定める。

$$\mathrm{path}(v, v') = \{v \text{ から } v' \text{ へのパスに含まれる辺 }\}$$
$$\mathrm{lpath}(v, v') = \mathrm{path}(v, v') \setminus \{(u, h(u)) \mid u \in V(T)\}$$

Lemma

任意の $v \in V(T)$ に対して

$$|\mathrm{lpath}(r, v)| \leq \log_2 w(r) - \log_2 w(v)$$

が成り立つ。つまり、根 r から v へのパスに含まれる「軽い辺」の個数が $\log_2 w(r)$ で抑えられる。

[証明のアイデア](Hopcroft のトリック): 頂点 $u \in V(T)$ とその子 u' について、u' が重い子 $h(u)$ でなければ

$$2 \cdot w(u') \leq w(u)$$

である。

縦 × 横 = 横 × 縦

根付き木 T と重み関数 w、辺の部分集合 $S \subseteq E(T)$ とする。

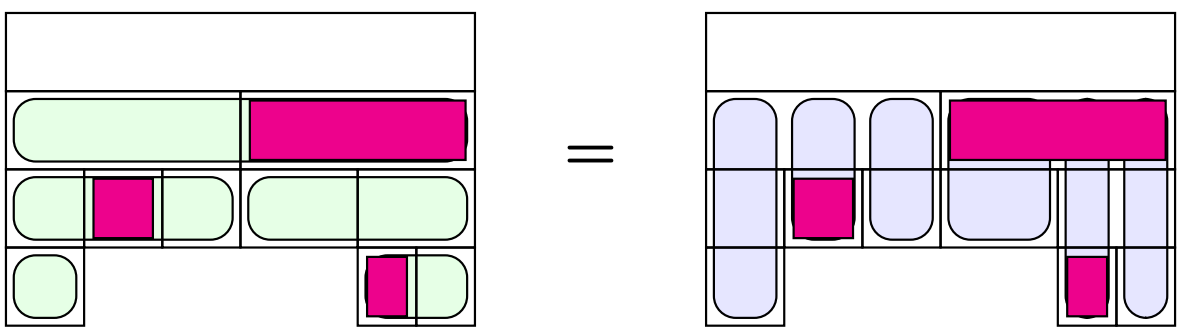
Lemma

次が成り立つ。

$$\sum_{v \in V(T)} \sum_{\substack{u \in \mathrm{ch}(v) \\ (v,u) \notin S}} w(u) \geq \sum_{T \text{ の葉 } l} |\mathrm{path}(r, l) \setminus S| \cdot w(l)$$

等号は重み関数がタイトなときに成立する。

[証明] 木の大きさに関する帰納法。
[直観的な証明]



タイト化

Lemma

根付き木 T と重み関数 w、重い子の指定 h があるとする。タイトな重み関数 w' であって、h が w' の重い子の指定であるようなものが存在する。

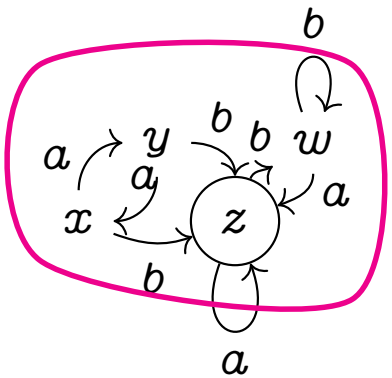
これまでの 3 つの観察を組み合わせて

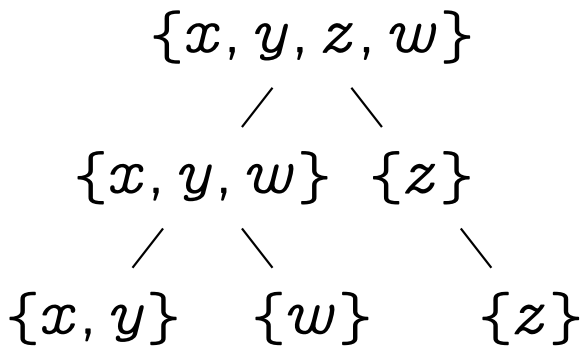
$$\sum_{v \in V(T)} \sum_{\substack{u \in \mathrm{ch}(v) \\ u \neq h(v)}} w(u) \leq w(r) \log_2 w(r) + \sum_{T \text{ の葉 } l} w(l) \log_2 w(l)$$

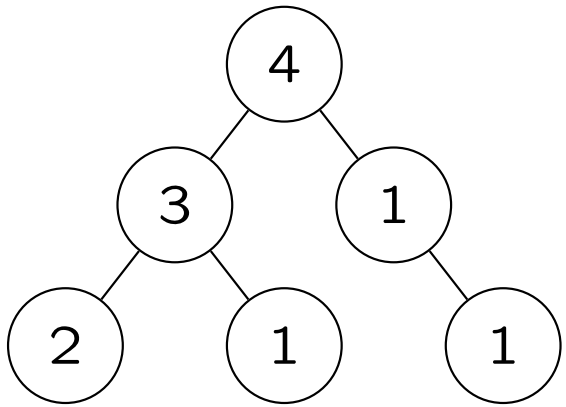
が証明できる。

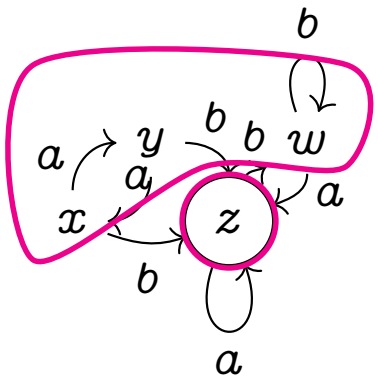
分割細分アルゴリズムの計算量の評価

細分の様子は重み付き木で表されるのであった。



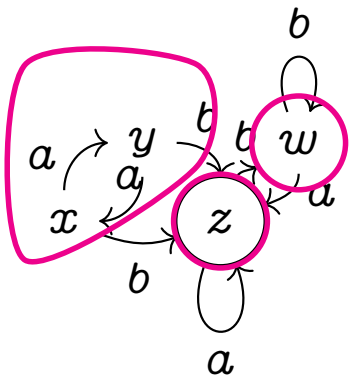






頂点 v に相当する同値類を細分して新たな同値類（に相当する頂点）$u_0, u_1, \ldots, u_n$ を得るとする。u_0 が最も大きい同値類だとする。
この細分を 1 回実行するのにかかる時間が $\mathcal{O}(K \sum_{i=1}^{n} w(u_i))$ であるとする。このとき Hopcroft の不等式から、アルゴリズムを実行するのにかかる合計の時間は

$$\mathcal{O}(K \sum_{v} \sum_{i=1}^{n} w(u_i)) \leq \mathcal{O}(Kn \log_2 n)$$



DFA に対する Hopcroft の分割細分アルゴリズム

重み付き木に対する Hopcroft の不等式

ファイブレーションを用いた分割細分アルゴリズム

余代数としての状態システム

$\mathbb{C}$ を圏とする。$F\colon \mathbb{C} \to \mathbb{C}$ を関手とする。
DFA $\mathcal{A}$ などの状態遷移システムは F 余代数 $c\colon C \to FC$ に一般化される。

システム	関手 $F\colon \mathbf{Set} \to \mathbf{Set}$	余代数 $c\colon C \to FC$
DFA	$FX = 2 \times X^{\Sigma}$	$c\colon C \to 2 \times C^{\Sigma}$
NFA	$FX = 2 \times \mathcal{P}(\Sigma \times X)$	$c\colon C \to 2 \times \mathcal{P}(2 \times C)$
Markov 鎖	$FX = \mathcal{D}(X)$	$c\colon C \to \mathcal{D}(C)$

一般の余代数に対して Hopcroft のアルゴリズムを考えたい。
特に計算量の評価をするために先ほどの木の議論を扱えるような形でアルゴリズムを記述したい。

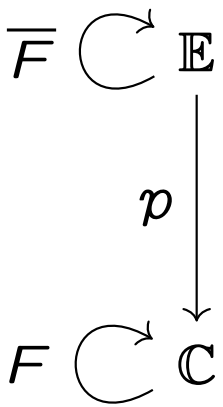
我々の貢献 (2): ファイブレーションを使って木の議論を扱えるような形でアルゴリズムを記述した

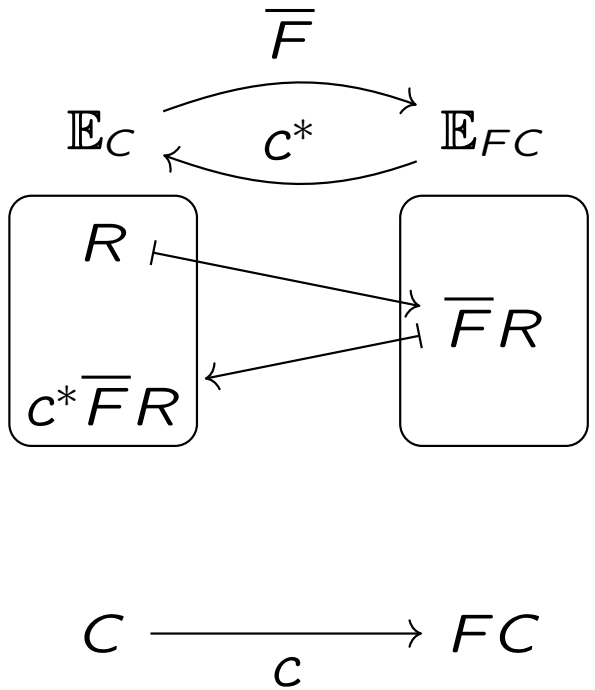
余代数の上の双模倣性関係

- $\mathbf{CLat}_\sqcap$ ファイブレーション $p\colon \mathbb{E} \to \mathbb{C}$
- 関手 $F\colon \mathbb{C} \to \mathbb{C}$
- F の持ち上げ $\overline{F}\colon \mathbb{E} \to \mathbb{E}$

余代数 $c\colon C \to FC$ に対してファイブレーションを用いて双模倣の概念を定義できる。[Hermida and Jacobs. 98]

$c^*\overline{F}\colon \mathbb{E}_C \to \mathbb{E}_C$ の最大不動点 $\nu(c^*\overline{F}) \in \mathbb{E}_C$ が C 上の「双模倣性関係」





例：NFA

- $\mathbf{CLat}_\sqcap$ ファイブレーション $p\colon \mathbf{EqRel} \to \mathbf{Set}$
- 関手 $F\colon \mathbf{Set} \to \mathbf{Set}$ は $FX = 2 \times \mathcal{P}(\Sigma \times X)$
- F の持ち上げ $\overline{F}\colon \mathbf{EqRel} \to \mathbf{EqRel}$ は、集合 C 上の同値関係 $R \in \mathbf{EqRel}_C$ に対して、次のように定める。

$$\overline{F}R = \left\{ \begin{array}{l} (\langle k, s\rangle, \langle l, t\rangle) \\ \in FC \times FC \end{array} \middle| \begin{array}{l} k = l, \text{ and for any } a \in \Sigma, \\ \forall x \in s(a).\exists y \in t(a).(x, y) \in R \\ \text{and} \\ \forall y \in t(a).\exists x \in s(a).(x, y) \in R \end{array} \right\}$$

すると C 上の同値関係 $\nu(c^*\overline{F}) \in \mathbf{EqRel}_C$ は通常の NFA の双模倣性関係と一致する。

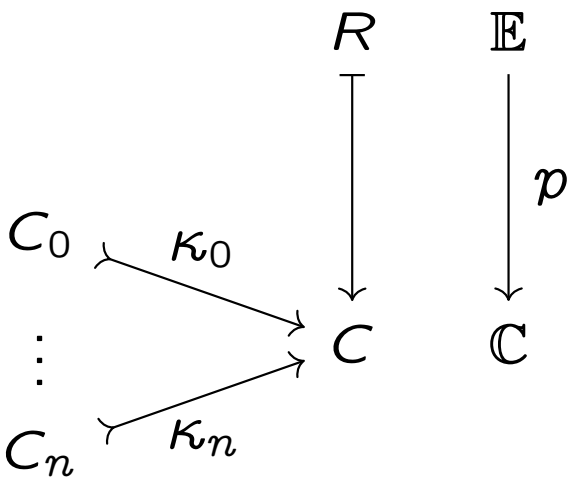
R 分割

「集合 C 上の同値関係 R の同値類 $\{C_i\}_{i\in I}$」をファイブレーションを使って定式化したい

- $\mathbf{CLat}_\sqcap$ ファイブレーション $p\colon \mathbb{E} \to \mathbb{C}$（$\mathbb{C}$ は十分余完備）
- $C \in \mathbb{C}$ と $R \in \mathbb{E}_C$

R 分割とは、C へのモノ射の族 $\{\kappa_i\colon C_i \rightarrowtail C\}$ であって、次を満たすもの

1. 任意の $i \in I$ に対して $\kappa_i^*(R) = \top_{C_i}$
2. $\bigsqcup_{i\in I}(\kappa_i)_*(\top_{C_i}) = R$
3. 異なる $i, j \in I$ に対して $C_i \cap C_j \cong 0$ かつ $C_i \ncong 0$



分割細分アルゴリズム $\mathrm{fPR}^{\mathrm{H}}$

- $\mathbf{CLat}_{\sqcap}$ ファイブレーション $p\colon \mathbb{E} \to \mathbb{C}$（$\mathbb{C}$ は十分余完備）
- 関手 $F\colon \mathbb{C} \to \mathbb{C}$ とその持ち上げ $\overline{F}\colon \mathbb{E} \to \mathbb{E}$
- $C \in \mathbb{C}$ と $R \in \mathbb{E}_C$
- C をコドメインとするモノ射 $\kappa\colon C' \rightarrowtail C$ に自然数 $w(\kappa) \in \mathbb{N}$ を割り当てる関数 w

が「よい条件」を満たしているとき、次のような計算を行うアルゴリズム $\mathrm{fPR}^{\mathrm{H}}$ を提案する。

入力： 余代数 $c\colon C \to FC$

出力： $\nu(c^*\overline{F})$ 分割 $\{\kappa_i\colon C_i \rightarrowtail C\}$

fPR$^{\mathsf{H}}$ の疑似コード

Input: A coalgebra $c\colon C \to FC$ in $\mathbb{C}$.
Output: A mono-sink $\{\kappa_i\colon C_i \rightarrowtail C\}_{i\in I}$ for some I.

$T := \{\epsilon\} \subset \mathbb{N}^*$; $C_\epsilon := C$; $C^{\mathsf{cl}}_\epsilon := 0$ ▷ initialisation
while There is $\rho \in L(T)$ such that $C^{\mathsf{cl}}_\rho \neq C_\rho$ **do** ▷ the main loop
  $R := \bigsqcup_{\sigma\in L(T)} (\kappa\sigma)_*(\top_{C_\sigma})$
  Choose a leaf $\rho \in L(T)$ such that $C^{\mathsf{cl}}_\rho \neq C_\rho$
  $R\rho := (c \circ \kappa_\rho)^*(\overline{F}(R))$
  if $R\rho = \top_{C_\rho}$ **then**
    $C^{\mathsf{cl}}_\rho := C_\rho$
    continue
  Take an $R\rho$-partitioning $\{\kappa_{\rho,k}\colon C_{\rho k} \rightarrowtail C_\rho\}_{k\in\{0,\dots,n_\rho\}}$
  Choose $k_0 \in \{0,\dots,n_\rho\}$ such as $w(C_{\rho k_0}) = \max_{k\in\{0,\dots,n_\rho\}} w(C_{\rho k})$
  MarkDirty(ρ, k_0)
  $T := T \cup \{\rho 0,\dots,\rho n_\rho\}$
return $\{\kappa\sigma\colon C_\sigma \rightarrowtail C\}_{\sigma\in L(T)}$
procedure MarkDirty(ρ, k_0)
  for $k \in \{0,\dots,n_\rho\}$ **do** $C^{\mathsf{cl}}_{\rho k} := C_{\rho k}$
  Let B be the pullback of the diagram:

$$\begin{array}{ccc} B & \rightarrowtail & C \\ \downarrow & \lrcorner & \downarrow c \\ F\left(C_{\rho k_0} \cup \left(\bigcup_{\sigma\in L(T)\setminus\{\rho\}} C_\sigma\right)\right) & \rightarrowtail & FC \end{array}.$$

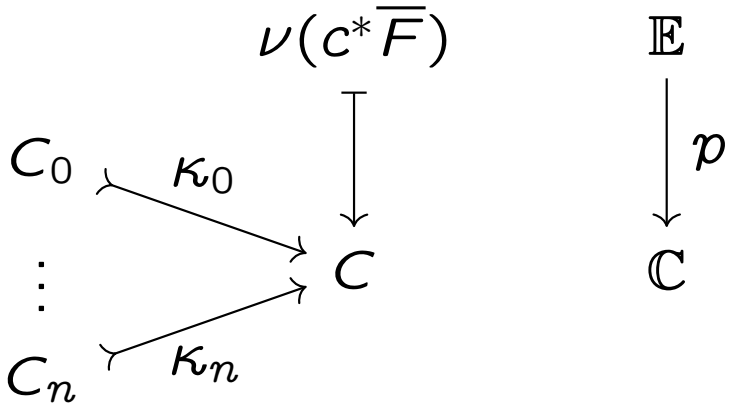
  for $\tau \in L(T \cup \{\rho 0,\dots,\rho n_\rho\})$ **do**
    $C^{\mathsf{cl}}_\tau := C^{\mathsf{cl}}_\tau \cap B$ ▷ states not in B are marked as dirty

主定理２（アルゴリズムの正しさ）

Theorem

- $\mathbf{CLat}_{\sqcap}$ ファイブレーション $p\colon \mathbb{E} \to \mathbb{C}$
- 関手 $F\colon \mathbb{C} \to \mathbb{C}$ とその持ち上げ $\overline{F}\colon \mathbb{E} \to \mathbb{E}$
- $C \in \mathbb{C}$ と $R \in \mathbb{E}_C$
- C をコドメインとするモノ射 $\kappa\colon C' \rightarrowtail C$ に自然数 $w(\kappa) \in \mathbb{N}$ を割り当てる関数 w

が「よい条件」を満たしているとき、$\mathsf{fPR}^{\mathsf{H}}$ は余代数 $c\colon C \to FC$ が入力されたとき、停止して、かつ $\nu(c^*\overline{F})$ 分割 $\{\kappa_i\colon C_i \rightarrowtail C\}_{i\in I}$ を出力する。



計算量の結果

前半で示した不等式から次のことが直ちにわかる。

Proposition

$\mathrm{fPR}^{\mathrm{H}}$ で各 MarkDirty の実行にかかる時間が $\mathcal{O}(K \sum_{k \in \{0,\ldots,n_\rho\} \setminus k_0} w(C_k))$ であるとする。このとき $\mathrm{fPR}^{\mathrm{H}}$ 全体での MarkDirty の実行時間の合計は $\mathcal{O}(K w(C) \log w(C))$ である。

$\mathrm{fPR}^{\mathrm{H}}$ のファイブレーション $p\colon \mathbb{E} \to \mathbb{C}$ を **EqRel** $\to$ **Set** で具体化して、重み関数 w として集合のサイズを与える関数 $w(C') = |C'|$ を採用すれば、Jacobs と Wißmann のアルゴリズムと本質的に同じアルゴリズムを得る。

Theorem

このようにして得られたアルゴリズムの計算量は入力された余代数 $c\colon C \to FC$ に対して $\mathcal{O}(f M |C| \log |C|)$ である。f は「一つの状態に対して遷移を計算するのにかかる時間」、M は c の「最大次数」。

[証明] 上の Proposition と償却計算量解析から従う。