yabaitech.tokyo vol.1

2018.10.08@ 技術書典 5

目次

RNN からのオートマトン学習 **..... 1**

MasWag

"Scheme の実装におけるスタックフレーム **(Draft)"**を実装する **..... 17**

wasabiz

自然な様相論理をつくる **..... 20**

zeptometer

あとがき **: SATΥSF$_{\mathrm{I}}$** で技術系同人誌を書いた話 **..... 38**

RNN からのオートマトン学習

MasWag

1. はじめに

本章では、ICML 2018 に採択された Extracting Automata from Recurrent Neural Networks Using Queries and Counterexamples [3] について、 オートマトン理論的な背景に重点を置いて解説します。 本章において、 形式言語理論についての知識は仮定しませんが、 素朴集合論や基本的なグラフ理論の知識や記号などは適宜使用します。 また、RNN についての基本的な知識を仮定します。 この論文では、 文字列の二値分類問題について学習済みの Long Short-Term Memory (LSTM) や Gated Recurrent Unit (GRU) などの Recurrent Neural Network (RNN) から Determistic Finite Automaton(決定的有限オートマトン、DFA) を抽出する手法を提案しています。 本手法は神託を用いて正規言語を学習する Angluin による L* アルゴリズム [1] を、RNN を用いて学習する手法に応用したものになります。L* アルゴリズムではある文字列 $w \in \Sigma^*$ が学習したい言語 $L \subseteq \Sigma^*$ に含まれているか ($w \in L$) を質問する**所属性質問** (membership query) と、学習した言語 $L' \subseteq \Sigma^*$ が学習したい言語 $L \subseteq \Sigma^*$ と等しいかどうかを質問する**等価性質問** (equivalence query) の二種類の質問を繰り返すことで効率良く、学習したい言語を最小の状態数を持つオートマトンで学習します。RNN を用いて学習する場合には、所属性質問に答えることは可能ですが (RNN に文字列を与えれば良い) 、等価性質問に答えることはとても難しいです。 この論文の主な貢献は、RNN を用いて (近似的に) 等価性質問に答える手法を与えた、ということになります。

本章の構成は以下のようになります。まず第二節で形式言語やオートマトンの必要な定義を与えます。本節では必要最低限の内容のみを扱います。より詳細は [2] などを参照してください。第三節では Angluin の L* アルゴリズムやそれに関連して Myhill-Nerode の

定理など正規言語の学習についての紹介を行います。第四節では本題の、RNN を用いて L* アルゴリズムの等価性質問に答える手法を説明します。

1.1. 記法

本章では実数の集合を $\mathbb{R}$、自然数の集合を $\mathbb{N}$ を用いて表わします。また、真理値を T と F を用いて表わします。また、数式中で変数及び単なる文字としてラテン文字が表われますが、原則として変数として用いるときには a の様に斜体で、文字として用いるときは a のように立体で表記します。

2. 形式言語とオートマトン

本章では有限集合 Σ を**文字**の集合として用います。例えば電話番号の様な数字の列を考える場合、Σ としてアラビア数字の集合 $\{0,1,2,3,4,5,6,7,8,9\}$ を用います。文字の有限列を**文字列**と定義します。文字列 w, w' を結合するときには演算子 $\cdot$ を用いて $w \cdot w'$ と表記します。例えば文字の集合 Σ が $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ のとき、110 や 0120444444 などは文字列となります。これらを結合した $110 \cdot 0120444444$、つまり 1100120444444 も文字列です。また、長さ 0 の特別な文字列として ε を用います。任意の文字列 w について ε は $w \cdot \varepsilon = \varepsilon \cdot w = w$ の性質が成り立ちます。以後長さ n の文字列 $a_1 a_2 \cdots a_n$ の集合を Σ^n、有限長についての集合和を $\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^n$ と書きます。また、文字の集合として以後主に $\Sigma = \{\mathrm{a},\mathrm{b}\}$ を用います。

文字列の集合を**言語**と言います。例えば文字の集合 Σ が $\Sigma = \{0,1,2,3,4,5,6,7,8,9\}$ のときに Σ^* の部分集合である、3 桁の電話番号の集合やフリーダイアルの電話番号の集合などは言語の例です。偶数の集合や回文となっている文字列の集合も言語の例です。

決定的有限オートマトン (Deterministic Finite Automata, DFA) は、様々な言語の中でも**正規言語**と呼ばれる言語を表現する方法の中の一つです。DFA は例えば図 1 にある、$\mathcal{A}$ のような状態遷移図を用いて表されます。DFA$\mathcal{A}$ が表現する言語を $L(\mathcal{A})$ と表記することとし、文字列 $w \in \Sigma^*$ が $L(\mathcal{A})$ に含まれるとき、$\mathcal{A}$ が w を**受理する**と言います。図 1 にある DFA$\mathcal{A}$ は、文字集合 $\{\mathrm{a},\mathrm{b}\}$ の上で定義された DFA で、「ab という部分列が存在する文字列」を受理します。この DFA には状態が q_0, q_1, q_2 の 3 つありますがその中に特別な役割をする状態が二種類あります。一つ目は**初期状態**と呼ばれる状態で、一つの

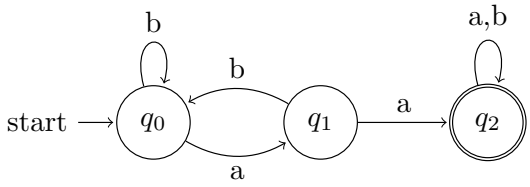


図 1 DFA の例 $\mathcal{A}$ 。"aa"という部分列を含む文字列を受理する。

DFA につき常に一つ存在します。 本章では初期状態は"start"と書かれた矢印で外から指すことで表すこととします。今回の初期状態は q_0 です。二つ目は**受理状態**と呼ばれる状態で、一つの DFA につき幾つ存在しても構いません。本章では受理状態は二重丸で表すこととします。今回の初期状態は q_2 です。

例として $\mathcal{A}$ が文字列 $w = \text{baab}$ を受理する様子を見ていきます。DFA ではまず始めに初期状態 q_0 にいるところから始めて、一文字ずつ文字を読んでいき、それに従って次の状態に進んでいきます。 まず始めに一文字目の b を読みます。 このとき、q_0 が始点で b のラベルが付いている辺の終点は q_0 なので、 次の状態も q_0 となります。 同様に二文字目の a を読むと、q_0 が始点で a のラベルが付いている辺の終点は q_1 なので、 次の状態は q_1 となります。 更に三文字目の a を読むと、q_1 が始点で a のラベルが付いている辺の終点は q_2 なので、 次の状態は q_2 となります。 最後に四文字目の b を読むと、q_2 が始点で b のラベルが付いている辺の終点は q_2 なので、最後の状態も q_2 となります。全ての文字を読み込んだ後で、現在居る状態が受理状態であるかどうかを調べます。今回の最後にいる状態 q_2 は二重丸の付いている受理状態なので、DFA$\mathcal{A}$ は $w = \text{baab}$ を受理する、 ということがわかります。同様の手順を踏むことで、任意の文字列 $w' \in \{\text{a,b}\}^*$ について $\mathcal{A}$ が w' を受理するかどうかを調べることができます。

DFA の形式的な定義は以下の様になります。

定義 1 (決定的有限オートマトン). 文字集合 Σ 上の決定的有限オートマトン $\mathcal{A}$ は 5 つ組 $\left(\Sigma, Q, q_0, Q_F, \Delta\right)$ である。但し Σ は有限の文字集合、Q は有限の状態集合、$q_0 \in Q$ は初期状態、$Q_F \subseteq Q$ は受理状態の集合、$\Delta : Q \times \Sigma \to Q$ は状態遷移関数である。

文字列 $w = a_1 a_2 \cdots a_n \in \Sigma^*$ に対して、q_i を各 $i \in \{1, 2, ..., n\}$ について帰納的に

$q_i = \Delta\big(q_{i-1}, a_i\big)$ と定める。$q_n \in F$ が成り立つとき、$\mathcal{A}$ は w を**受理する**という。また、$\mathcal{A}$ が受理する文字列の集合を $\mathcal{A}$ の**受理言語**といい、$L(\mathcal{A})$ と表記する。 言い換えると $L(\mathcal{A}) = \Big\{a_1 a_2 \cdots a_n \mid q_n \in F, \text{where } q_i = \Delta\big(q_{i-1}, a_i\big)\Big\}$ である。また、DFA$\mathcal{A}$ が言語 $L(\mathcal{A})$ を**認識する**、という。DFA によって受理される言語を**正規言語** (regular language) という。以後文字列 $w = a_1 a_2 \cdots a_n$ について、$\Delta(q, w) = \Delta\Big(\Delta\big(\cdots\Delta\big(q, a_1\big), \cdots, a_{n-1}\big), a_n\Big)$ と定める。

2.1. 余談

上で正規言語を定義しましたが、 言語全体で見ると勿論正規言語以外の言語も多数あります。というより、正規言語は DFA という**有限状態**の遷移図で書き表せるという点でかなり制限が強い言語であり、 一方で様々な良い性質を持った言語です。 例えば次節で紹介する L* アルゴリズムの停止性は、正規言語の"有限状態らしさ"による特徴付けである、Myhill-Nerode の定理と深い関係があります。

3. L* アルゴリズム

前節で定義した正規言語を DFA の形式で人間が書くことも当然できますが、これ以降本章では未知の正規言語を学習するという問題を主に考えます。 機械学習の教師付き二値分類問題では「正例と負例からなる訓練用データを与えて、 訓練用以外のデータについても高い確率で正しく分類する」という問題設定が良く用いられますが、ここでは exact learning と呼ばれる学習を考えます。Exact learning では、 未知の正規言語を近似する正規言語を学習するのではなく、 未知の正規言語と完全に等しい正規言語を表現する DFA を学習します。 ここで、 任意の文字列の有限集合は正規言語であるので、 有限個の訓練用データを与えても本当に学習したい正規言語に辿り着くのは難しそうです。 本節では (というより exact learning の文脈ではしばしば) 、訓練用データの代わりに次の二つの質問を神託に聞くことを通して学習を行なう、 という設定を考えます。 現実的にこんなことがわかる神託をどうやって用意するのか、という問題はここでは考えません。

- 所属性質問 (membership query): 文字列 $w \in \Sigma^*$ を神託に与えて、w が学習したい言語 $L \subseteq \Sigma^*$ に含まれているか ($w \in L$) を問う質問。

- 等価性質問 (equivalence query): 正規言語 $L' \subseteq \Sigma^*$ を神託に与えて、L' が学習したい言語 L と等しいかどうかを問う質問。$L' \neq L$ である場合には反例 $w \in \Sigma^*$、つまり $w \in L \triangle L'$ を充たす最短の文字列 w が返る。ここで $\triangle$ は集合の対称差の記号である。

本節では例として $L = \{a \in \{\text{a,b}\} \mid \text{a, b の個数が共に偶数}\}$ を用います。例えば図 2 の DFA がこの言語を認識します。

3.1. 観察表

L* アルゴリズムでは**観察表** (observation table) と呼ばれる表 T に、各質問の結果を随時記入していくことで学習を進めていきます。観察表は図 3 にある様に行、列が共に Σ 上の文字列で添字付けされている表です。観察表の各セルには所属性質問の結果が記入されます。各セル $T[w, w']$ には文字列 $w \cdot w'$ の所属性質問の結果が記入されます。また、観察表には水平線より上側と下側の区別があります。水平線より上側の行の添字集合を $P \subseteq \Sigma^*$, 下側の添字集合を $P' \subseteq \Sigma^*$, また、列の添字集合を $S \subseteq \Sigma^*$ と呼びます。また、観察表が以下の条件を充たすとき、その観察表を**閉じている**と言います。

- 各 $w' \in P'$ について、$T[w, \text{-}] = T[w', \text{-}]$ を充たす $w \in P$ が存在する
- 各 $w' \in P$、$a \in \Sigma$ について、$w \cdot a \in P \cup P'$ が成り立つ

3.2. L* アルゴリズム

L* アルゴリズムではまず始めに図 4 の観察表から始めます。最初に空欄になっているセル $T[\varepsilon, \varepsilon]$ を所属性質問を用いて埋め、同時に P' に a,b を追加し、同様に埋めます (図 5) 。

ここで図 5 の観察表が閉じているかを判定します。今回は任意の $w \in P$ について、行 a の行ベクトル $T[\text{a,-}]$ が行 w の行ベクトル $T[w, \text{-}]$ と一致しないので閉じていません。そこで閉じていない原因の行 a を P に追加します。このとき各 $\{\text{a} \cdot a \mid a \in \Sigma^*\}$ を添字と

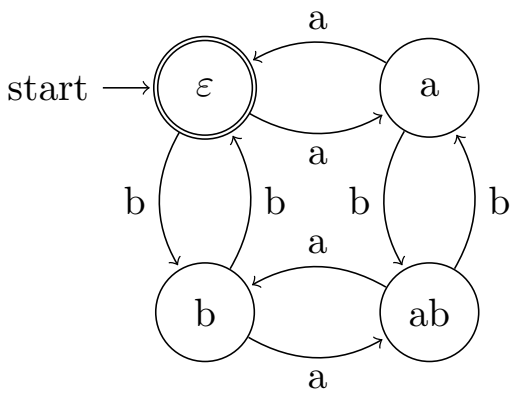


図 2 a, b の個数が共に偶数の文字列の言語を認識する DFA の例。

	ε	a
ε	T	F
a	F	T
b	F	F
aa	T	F
ab	F	F
ba	F	F
bb	T	F

図 3 観察表の例。$P = \{\varepsilon, \text{a,b}\}, P' = \{\text{aa,ab,ba,bb}\}, S = \{\varepsilon, \text{a}\}$

	ε
ε	

図 4 一番始めの観察表

	ε
ε	T
a	F
b	F

図 5 ε 、a 、b について埋めた観察表

する列が P と P' のどちらにもに存在しない場合、その存在しない添字を P' に追加します (図 6) 。

	ε
ε	T
a	F
b	F
aa	T
ab	F

図 6 ε 、a 、b について埋めた後、閉じるまで操作を続けた観察表

今回は観察表が閉じました。観察表が閉じている場合、次のルールに従って観察表から DFA を生成することができます。

- 各 $w \in P$ に対して、DFA の状態 q_w を生成する
- DFA の初期状態は q_ε とする
- DFA の受理状態は $T[w, \varepsilon] = \mathrm{T}$ となる状態 q_w とする
- DFA の遷移関数 $\Delta\left(q_w, a\right)$ は、$w \cdot a \in P$ の場合 $\Delta\left(q_w, a\right) = q_{w \cdot a}$ とし、そうでない場合 $T[w \cdot a, \text{-}] = T[w', \text{-}]$ を充たす $w' \in P$ について、$\Delta\left(q_w, a\right) = q_{w'}$ とする。

例えば図 6 の観察表からは図 7 の DFA 、$\mathcal{A}_1$ が生成されます。

DFA を一つ生成することができたので等価性質問によってこの DFA 学習したい正規言語を認識するかどうかを調べてみます。今回、DFA$\mathcal{A}_1$ は言語 L を認識しないので、最短の反例 ba が返ります。次に反例 ba がどの位置から図 6 の観察表と食い違ったかを見てみます。今回は T[b,-] $= T$[a,-] で、実際 $b \notin L$ と $a \notin L$ が成り立ちますが、$ba \notin L$ 、$aa \in L$ となるので末尾の a で観察表と食い違いました。 これはつまり現在の状態の受理・非受理のみではなく、現在の状態から a を読んだときの受理・非受理も考慮する必要があるということになります。これを表わすために S の列に a を加え、先と同様に観察表が閉じるまで所属性質問を繰り返します (図 8) 。

図 8 の下側の観察表から生成される DFA について等価性質問を問い合わせると学習したい言語と一致するので、 今回の学習はここで終了となります。 ここまでの流れだけを見ると今回たまたま上手く行っただけで、 この学習が有限回で停止しない場合もあるの

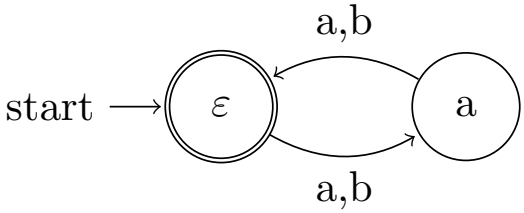


図 7 図 6 の観察表に対応する DFA$\mathcal{A}_1$

	ε	a
ε	T	F
a	F	T
b	F	F
aa	T	F
ab	F	F
ba	F	F
bb	T	F

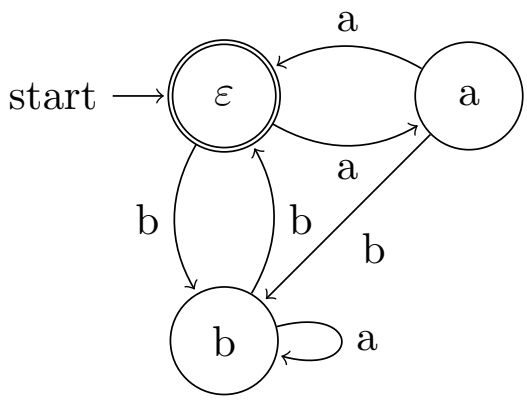
	ε	a	b
ε	T	F	F
a	F	T	F
b	F	F	T
ab	F	F	F
aa	T	F	F
ba	F	F	F
bb	T	F	F
aba	F	F	T
abb	F	T	F

図 8 操作を更に続けた観察表

ではないかと疑問に思うと思われますが、実は常に有限回 (より正確には多項式回) で停止しますし、更にはこの方法で生成される DFA は与えられた言語を認識する DFA のなかで状態数が最小となることもわかります。これらの点について次に説明していきます。

3.3. Myhill-Nerode の定理と DFA の最小性

さて、L* アルゴリズムの停止性を見るために、まずは正規言語の「有限らしさ」の特徴



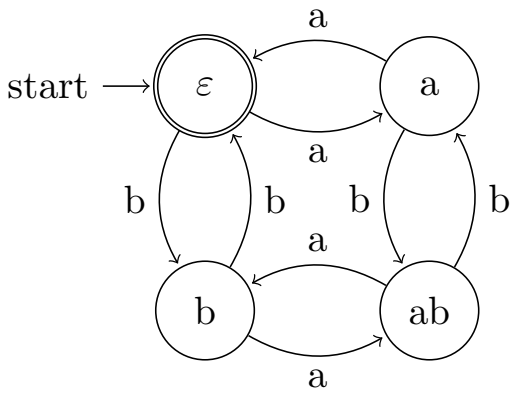


図 9 図 8 の観察表に対応する DFA$\mathcal{A}_2$ 及び $\mathcal{A}_3$

付けである Myhill-Nerode の定理を見ていきます。

定義 1 (Myhill-Nerode 同値関係). 有限文字集合 Σ 上の言語 L について、Myhill-Nerode **同値関係** $R_L \subseteq \Sigma^* \times \Sigma^*$ を、次のように定義する。文字列 $w, w' \in \Sigma^*$ について wR_Lw' が成り立つのは以下のときである。

$$\forall w'' \in \Sigma^*.w \cdot w'' \in L \Leftrightarrow w' \cdot w'' \in L$$

定理 2 (Myhill-Nerode の定理). 有限文字集合 Σ 上の言語 L について、L が正規言語であることの必要十分条件は、商集合 L/R_L が有限集合となることである。

文字列 w, w' と Myhill-Nerode 同値関係 R_L について、wR_Lw' が成り立つということの直観は、「これから与えられる未知の文字列 $w'' \in \Sigma^*$ について $w \cdot w'' \in L$ が成り立つかどうかを調べたいときに、今まで読んだ文字が w であるか w' であるかは気にしなくて良い」ということになります。このことについて木を使ってより詳しく見ていきます。文字集合 $\Sigma = \{\text{a,b}\}$ 上の言語 L として、「a と b の出現回数が共に偶数回」というものを考えます。この言語を受理する (無限状態数の) オートマトンとして図 10 の木を考えます。この木では各文字列 $w \in \Sigma^*$ について一つの状態が割り当てられ、L に含まれる文字列と対応する状態が受理状態となります。ここで今回、例えば $\text{b}R_L\text{aba}$ が成り立ちますが、これは木の上の性質としてみると、b を根としたときの部分木と、aba を根としたときの部分木が等しいということになります。従って図 11 の (無限状態の) オートマトンのように、状態 b と状態 aba を一つにまとめても、受理言語は変わらないと言えます。商集合 L/R_L が有限集合、つまり右側のオートマトンのように状態を纏める操作を行うと最終的に有限状態のオートマトンになるとき、確かに $L(\mathcal{A}) = L$ が成り立つような、状態数 $\left|L/R_L\right|$ の DFA$\mathcal{A}$ を構成できるので L は正規言語になります。また、L が正規言語である、つまり $L(\mathcal{A}) = L$ が成り立つような DFA$\mathcal{A}$ が存在するとき、文字列 w, w' について $\delta\left(q_0, w\right) = \delta\left(q_0, w'\right)$ が成り立つならば wR_Lw' も成り立つので L/R_L は有限集合となります。以上が Myhill-Nerode の定理が成り立つ簡単な理由です。さらに、正規言語 L と DFA$\mathcal{A}$ について、状態数が $\left|L/R_L\right|$ より少ない場合 $\delta\left(q_0, w\right) = \delta\left(q_0, w'\right)$ であって wR_Lw' が成り立たないような文字列 $w, w' \in \Sigma^*$ が存在するので、次の系が成り立ちます。

系 3 (状態数最小の DFA)**.** 正規言語 L について、状態数 $\left|L/R_L\right|$ で L を認識する *DFA* が存在し、また、状態数 L/R_L 未満の任意の *DFA* は L を認識しない。

さて、Myhill-Nerode の定理の立場から L* アルゴリズムの観察表を見ていきます。まず、各 $w \in P$ について行ベクトル $T[w, \text{-}]$ は互いに異なる、つまり適宜末尾に文字列を加えたときの受理 / 非受理の関係が異なるので、P の要素数は L/R_L の要素数以下になり、L が正規言語であるとならば上限が存在します。各 P' の要素は P の要素に一文字付け加えたものであるので、$\left|P'\right| \leq |P| \times |\Sigma|$ となります。また、S に添字 w' を追加する際は w' を追加することで新たに食い違うような w' のみを追加する、つまり各 $w \in S$ について $T[u, w] = T[u', w]$ であるが $T[u, w'] \neq T[u', w']$ である様な $u, u' \in P \cup P'$ が存

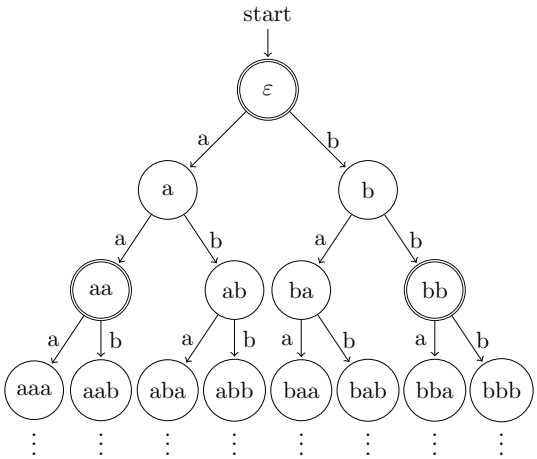


図 10 a, b の個数が共に偶数の文字列の言語を認識する無限状態のオートマトンの例

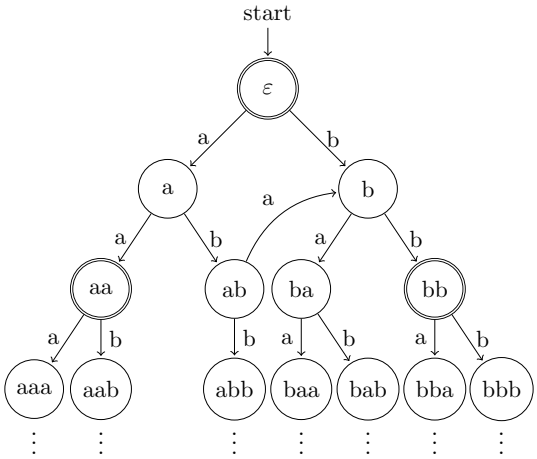


図 11 図 10 の無限状態オートマトンの状態 a と aba を纏めたオートマトン

在するときのみ w' を S に追加します。従って、$\left|S\right| \leq \left|L/R_L\right| \times \left(1 + \left|\Sigma\right|\right)$ も成立します。更に観察表を閉じる際及び等価性質問の結果反例が返って来た際には P と S の大きさが 1 以上増えます。以上より L* アルゴリズムは L が正規言語であれば常に停止し、$L\left(\mathcal{A}\right) = L$ を充たす DFA$\mathcal{A}$ を返すことがわかります。また、L* アルゴリズムが生成する DFA の状態数は $\left|P\right|$ ですが、$L\left(\mathcal{A}\right) = L$ を充たす任意の DFA の状態数は L/R_L 以下であることから、$\mathcal{A}$ は $L\left(\mathcal{A}\right) = L$ を充たす状態数が最小の DFA であることもわかり

ます。

4. RNN による等価性質問

前節では所属性質問と等価性質問に答える神託が存在する、という仮定の元で正規言語を学習する、L* アルゴリズムについて説明しました。本節ではこのような神託を RNN を用いて近似的に実現する手法について説明します。本節の内容が [3] の主な貢献となります。本節での目標は、正規言語 L の受理 / 非受理についての二値分類問題を正しく解く RNN R が与えられた際に、前節で考えていた所属性質問と等価性質問に答える、というものになります。このような問題を既存の RNN の学習手法及び前節で説明した L* アルゴリズムと組み合わせることにより、例えばノイズの乗った訓練用データから正規言語を学習することができます。

本節では、RNN の状態空間を $\mathbb{R}^N$、RNN の初期状態ベクトルを $h_0 \in \mathbb{R}^N$、RNN の状態ベクトル $h \in \mathbb{R}^N$ に文字 $a \in \Sigma$ ないし文字列 $w \in \Sigma^*$ を与えたときの状態ベクトルを $g\big(h, a\big)$ や $g\big(h, w\big)$、RNN の状態 $h \in \mathbb{R}^N$ に対して受理、非受理の二値を割り当てる関数を $f : \mathbb{R}^N \to \{\mathrm{T}, \mathrm{F}\}$ と表記します。つまり、文字列 $w \in \Sigma^*$ について、$f\Big(g\big(h_0, w\big)\Big) = \mathrm{T}$ であるとき、この RNN は文字列 w を受理すると分類した、ということになります。

さて、まず所属性質問について RNN を用いて答えることを考えますが、これはとても簡単です。というのも上記の様に、文字列 $w \in \Sigma^*$ について、$f\Big(g\big(h_0, w\big)\Big)$ の値を見ることで w が学習したい言語 L に所属しているかどうかを調べることができるからです。従って以下では RNN を用いて等価性質問に如何に答えるかということに注目していきます。

4.1. RNN による等価性質問の概要

等価性質問について、[3] では RNN からある種のオートマトンを構成しながら答えていきます。まず、状態数が無限状態で構わなければ、状態空間を $\mathbb{R}^N$、初期状態を h_0、受理状態を $f^{-1}(\mathrm{T})$、遷移関数を $\Delta\big(h, a\big) = g\big(h, a\big)$ とすることで、RNN を無限状態のオートマトンと見ることができます (図 12)。しかし、等価性質問に答えるためには状態数が有限である必要があります。そこで、Myhill-Nerode 同値関係のように、同じ状態と見な

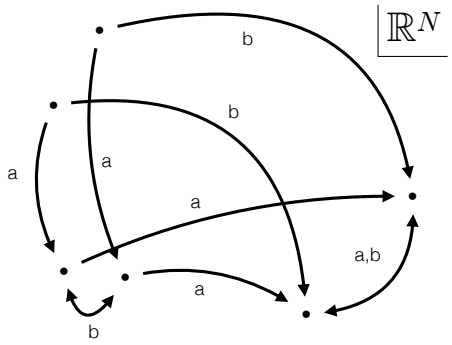


図 12 RNN を無限状態のオートマトンとして見た例

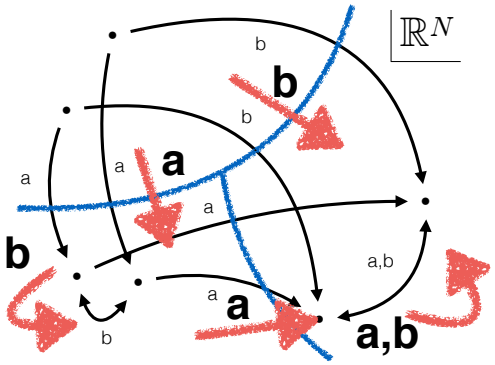


図 13 図 12 の RNN からクラスタリングによって DFA を構築した例

しても構わない状態を一つのクラスタとして纏める (クラスタリング) ことで DFA を構築します (図 13)。つまりクラスタリングの関数 $p : \mathbb{R}^N \to \mathbb{N}$ で、以下の性質が成り立つものを求めることができれば、p の像を状態空間とする DFA を構築することができます。

- RNN の状態 $h, h' \in \mathbb{R}^N$ について $p(h) = p(h')$ ならば任意の文字 $a \in \Sigma$ について $p\big(g(h,a)\big) = p\big(g(h',a)\big)$
- RNN の状態 $h, h' \in \mathbb{R}^N$ について $p(h) = p(h')$ ならば $f(h) = f(h')$

- クラスタリング p の像 $\left\{p(h) \;\middle|\; h \in \mathbb{R}^N\right\}$ が有限集合である

ところで、このクラスタリングの関数 p を正確に求めることができるのであればそもそも RNN から DFA を抽出できているので、DFA が正確に学習できているとも言えますが、それは非常に難しいです。例えば矛盾が起こらない様に RNN の全ての状態について幅優先探索によって調べれば正確なクラスタリングを行うことができますが有限時間では行えません。従ってこのクラスタリングについては途中で諦める必要があります。この途中で諦めるという点について、Myhill-Nerode の定理などを使いつつ、程良く諦める手法を提案しているというのが [3] の技術的な肝になります。以下では RNN による等価性質問の具体的な流れについて説明をします。

4.2. 探索のための初期化

先に説明した様に [3] では幅優先探索によってクラスタリングを求めます。おおまかな流れは、入力文字列について探索することによって、今のクラスタリングにおける反例を見付け、それに基いてクラスタリングを更新する、という操作をクラスタリングが十分正確になるまで繰り返すことで、十分正確なクラスタリングを得るというものになります。クラスタリング p は任意の $h \in \mathbb{R}^N$ について $p(h) = 0$ となるように、次に調べる文字列のキュー W_{new} に ε を push した状態で初期化し、訪問済の文字列の集合 $W_{\text{seen}} \in \mathcal{P}(\mathbb{N})$ 及びクラスタの集合 $P_{\text{seen}} \in \mathcal{P}(\mathbb{N})$ は 共に $\emptyset$ で初期化します。

4.3. Step 1: 矛盾かどうかの確認

まず始めに次に調べる文字列のキュー W_{new} から文字列 w を pop します。ここで既にキュー W_{new} が空であった場合、入力の RNNR と DFA$\mathcal{A}$ は十分似ているということで、等価であるという返答を返します。

文字列 w について、まず RNNR における受理 / 非受理と DFA$\mathcal{A}$ における受理 / 非受理が一致するかを調べて、一致しない場合は w を反例として返します。これによって、既に調べた文字列については R と $\mathcal{A}$ が同じ分類をするということを保証できます。次に、今まで調べた文字列 w' で、$p\big(g\big(h_0, w\big)\big)$ と $p\big(g\big(h_0, w'\big)\big)$ が一致するものについて、DFA

$\mathcal{A}$ 上の状態 $\Delta\big(q_0, w\big)$ と $\Delta\big(q_0, w'\big)$ が一致するかどうかを調べます。これらが一致する場合、クラスタリングに矛盾はないので Step 2-1 に移ります。これらが一致しない場合クラスタリングに問題があるか R と $\mathcal{A}$ が一致しないということになるので、Step 2-2 でより詳しく見ていきます。

4.4. Step 2-1: 新たな点の追加

DFA$\mathcal{A}$ 上の状態 $\Delta\big(q_0, w\big)$ と $\Delta\big(q_0, w'\big)$ が一致する場合、まず w を訪問済の文字列の集合 W_{seen} に追加します。$p(w)$ が既に訪問済である (つまり $p(w) \in P_{\text{seen}}$) 場合はそのまま Step 1 に戻りますが、$p(w)$ に未だ訪問したことがない場合、w に一文字加えた文字列についても探索を行います。つまり、$p(w)$ を P_{seen} に追加して、各 $a \in \Sigma$ について $w \cdot a$ を W_{new} に push して、Step 1 に戻ります。

4.5. Step 2-2: クラスタリングの改良

DFA$\mathcal{A}$ 上の状態 $\Delta\big(q_0, w\big)$ と $\Delta\big(q_0, w'\big)$ が一致しない場合、この不一致の原因がクラスタリングにあるのか、そもそも RNNR と DFA$\mathcal{A}$ の表している言語が異なるのかを調べる必要があります。そのために、まず $\Delta\big(q_0, w \cdot w''\big)$ と $\Delta\big(q_0, w' \cdot w''\big)$ が異なる様な文字列 $w'' \in \Sigma^*$ を見付けます。このような文字列は L* アルゴリズムによって得られる DFA の最小性から必ず存在します。

次に、各 $p\big(g\big(h_0, w\big)\big) = p\big(g\big(h_0, u\big)\big)$ を充たすような訪問済の文字列 $u \in W_{\text{seen}}$ について、RNNR と DFA$\mathcal{A}$ において $u \cdot w''$ の受理 / 非受理が一致しないものが存在するかどうかを調べます。もし受理 / 非受理が一致しないものが存在するのであれば、それが反例となるので、R と $\mathcal{A}$ が異なる言語を認識するという証拠として $u \cdot w''$ を返します。もし受理 / 非受理が常に一致するのであれば、本来 $p\big(g\big(h_0, w\big)\big) \neq p\big(g\big(h_0, w'\big)\big)$ であるべきであるにもかかわらず、クラスタリングが粗すぎるために $p\big(g\big(h_0, w\big)\big) = p\big(g\big(h_0, w'\big)\big)$ であると言うことができます。そのため、SVM などによってクラスタリングを改良して、Step 1 に戻ります。このとき、今迄訪問した点についての情報を引き継ぐことができないので、W_{new}、$W_{\text{seen}} \in \mathcal{P}(\mathbb{N})$、$P_{\text{seen}} \in \mathcal{P}(\mathbb{N})$ は全て初期化します。

5. 結論

実験結果や疑似コードなどは元の論文に譲りますが、本章で説明した手法によって

RNN を用いて L* アルゴリズムの神託を実装することで DFA を学習することができます。 この論文の成果は、 最近の機械学習の発展と古典的な形式言語理論の結果を組み合わせたという理論的な点のみならず、 基本的にパラメタチューニングの必要がないため RNN さえ得られれば容易に DFA を抽出できるという特徴もあります。 また、 例えば正規言語ではない言語を学習させた RNN に対して L* アルゴリズムを用いた際にどのような挙動をするのか、など更に興味深い結果が今後得られるのではないでしょうか。

参考文献

[1] Dana Angluin. Learning Regular Sets from Queries and Counterexamples. *Inf. Comput.*, 75(2), pages 87–106, 1987.

[2] J. ホップクロフト , R. モトワニ , J. ウルマン . オートマトン言語理論計算論 *I*. サイエンス社 , 2003.

[3] Gail Weiss, Yoav Goldberg, Eran Yahav. Extracting Automata from Recurrent Neural Networks Using Queries and Counterexamples.. In *ICML*, pages 5244–5253, 2018.

"Scheme の実装におけるスタックフレーム (Draft)"を実装する

wasabiz

1. はじめに

この章では，「Scheme の実装におけるスタックフレーム (Draft)」というウェブページに掲載されているアイデアを基に Scheme のコンパイラを作成していきます．Scheme は私の個人的なお気に入りのプログラミング言語の一つで，これまでもいくつか処理系を作成してきました．その過程で Scheme の実装方法についていくつもの文献を読んできました．中でも，件のサイトに掲載されている手法はシンプルでありながら効率的かつ巧妙で，Scheme の実装方法として最も美しい手法の一つであると言えるでしょう．これは，このページの著者である shiro さんが Gauche という有名な Scheme 処理系の作者であり，その長い実装の経験があった上で辿りついた結論だからなのかもしれません．しかしながら，Gauche 自体はこの手法に基づいて実装されているわけではなく，また他にこのページの内容に沿って実装されている処理系も私の知る限り存在しません．つまり，非常に魅力的な内容のアイデアが，実装されないまま何年も野晒しになっているという訳です．個人的にこの状況はとても勿体無いと感じながらも長いこと何もしないまま過ごしてきたのですが，今回この同人誌に参加するにあたってこれまでやってこなかった実装をやってみることにしました．

実装にあたっては C 言語を中間言語として用いることにしました．これは，レジスタ割り当てなどの手間を省くためと，もうひとつは私が単に C 言語が好きなためです．あとで出てくるのですが，この手法の難しいところはガベージコレクションです．しかし C 言語を経由して実装する場合，そこを中間言語のコンパイラの実装 (ここでは gcc) に丸投げすることが出来ません．また，Scheme のような動的型の言語を実装するために複数

の種類の型をひとつにして扱う必要がありますが，それについても C 言語で標準的な手法が用意されている訳ではありません．これらの問題に対処するため，この記事ではまず，C 言語のためのガベージコレクションライブラリと動的型付きプログラミング用ライブラリを作成するところから初めます．結果的に，C 言語を中間表現として用いる場合の困難さが大きく提言され，メンテンナンスが容易でかつ効率的なコンパイラを実装することができます．

以下ではまず初めに，背景となる「Scheme の実装におけるスタックフレーム (Draft)」の内容について解説したあと，関連する話題として「Cheney on the M.T.A.」と呼ばれる Scheme の実装テクニックについて説明します．「Cheney on the M.T.A.」は Chicken Scheme 等の実装で使われている手法で今回実装する手法と多くの共通点を持ちます．さらに，実装の細かなテクニックとして，C 言語でガベージコレクションと動的型付プログラミングを行うためのライブラリを作成します．これらのライブラリは今回のコンパイラ作成に便利であるだけでなく，通常の C 言語のプログラミングにも利用可能です．いずれもヘッダファイルのみで記述されているため，include するだけで使えます．その後にコンパイラの実装の詳細を紹介し，その評価を行います．さいごにまとめでこの章を締め括ります．

「"Schemeの実装のおけるスタックフレーム(Draft)"を実装する」休載のお知らせ

…という企画を考えていたのですが、入稿5日前に右手を骨折してしまい、記事の完成が不可能な状況となってしまいました。すでにコンパイラはおおよそ動いており、また予定していた6つのセクションのうち5つは書き終えているのでいつか完成させて印刷まで辿り着ければと思っています。

ところで、この数日で左手だけでトーキョーをサバイブしていくためのノウハウが結構溜まってきたのでそのうちまとめたいですね。

それでは。

yabaitech.tokyo

自然な様相論理をつくる

zeptometer

1. はじめに

この記事では Pfenning らによって提唱された Judgmental Reconstruction[4] の手法をもとにして直観主義様相論理の自然演繹の体系を作っていきます。 元の論文においてはこの手法を用いて直観主義の S4 様相論理を構築していますが、この記事では直観主義様相論理に関係するいくつかの体系 (Benton の随伴論理の亜種、Fitch-style の直観主義 K 様相論理、Dual-Context の直観主義 K 様相論理) を同様の手法で説明し、これらの体系を同じ枠組みで説明することを試みます。(以降、 単に様相論理と書いた場合直観主義様相論理のことを指します)

計算機科学及び論理学では Curry–Howard 同型と呼ばれる論理と計算の間の対応関係があることが知られています。Judgmental Reconstruction の面白い点は論理体系としての正当化をそのまま計算としてみなすことができるという点です。紙面の都合上この記事では論理体系としての側面のみについて記述しますが、それぞれの論理体系について素直に型付き λ 計算の体系が対応します。

尚、この記事は以下の背景知識を前提としています。

- 直観主義論理、自然演繹及び Curry–Howard 同型
- 様相論理の基礎 (Kripke 意味論を除く)

構成は以下のようになっています。2 節では Judgmental Reconstruction の概念を説明しながら例として直観主義論理の体系 IPC を構築します。3 節では IPC を拡張して二つの階

層を持つ論理 IPC2 を構築します。その後 4 節で IPC2 を更に拡張して K 様相論理の体系の IPC2M と IPCkM をそれぞれ構築します。最後に 5 節でまとめておしまいです。

2. 直観主義論理をつくる

Martin-Löf は彼の著作において [2] 論理体系における判断 (Judgment) を 「知の対象 (the object of knowledge)」 だと説明しています[*1]。例えばこの後に紹介する IPC という論理体系では $A_1,...,A_n \vdash B$ という判断は「$A_1,...,A_n$ の命題を全て仮定した時に B という命題を結論できる」という知識を表します。Judgmental Reconstruction とは判断の性質を土台として自然演繹の論理体系 (とそれに対応する型付き λ 計算の体系) を構築する方法論を指します。これがどのようなものかを理解してもらうために、$\rightarrow$ のみを論理結合子として含むような直観主義論理の体系 IPC(Intuitionistic Propositional Calculus) をつくってみることにします。 この流れは基本的に元の論文 [4] のそれと同じですが、 一部手を加えていることにご留意ください。

2.1. 判断の構造を決める

最初に全体の定義を図 1 に示しておきます。 記号の定義をした後に最初にすることは「判断の構造を決める」ことです。

IPC だと判断は $\Gamma \vdash A$ という形をしており、 これは 「Γ の命題を全て仮定した時に A が成り立つ」 ということを表しています。 判断に仮定という仕組みを入っている点が IPC の (見過ごされがちだけれども) 重要な点です。

この判断がどのような構造を持つか見てみましょう。まず、判断の意味から仮定のうちの一つをそのまま結論として導くことは至って妥当だと考えられます。これに対応するのが (hyp) の規則です。また、仮定について「余計な仮定を付け加えたり」「同じ仮定を一つに潰したり」「仮定の順番を変えたり」しても導くことのできる結論には特に変更はないように思われます。 この辺りの直観を構造規則として表したものが (weak)(contr)(exchg) の三つの規則で、 それぞれ弱化 (weakening) 、 縮約 (contraction) 、 交換 (exchange) と呼ばれます。 また、$\Gamma, A \vdash B$ が成り立っている時に A が Γ から導けるのであれば、 元の

1 このアイデアは実際に Judgmental Reconstruction のベースになっていて、Pfenning の論文 [4] でも言及されています。

記号の定義

命題変数	$p, q, \ldots$	
命題	A, B	$:= p \mid A \to B$
文脈	Γ, Δ	$:= \cdot \mid \Gamma, A$

判断の形

$$\Gamma \vdash A$$

仮定の使用

$$\frac{A \in \Gamma}{\Gamma \vdash A}\,(\text{hyp})$$

期待される構造規則

$$\frac{\Gamma \vdash B}{\Gamma, A \vdash B}\,(\text{weak}) \qquad \frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\,(\text{contr}) \qquad \frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, B, A, \Delta \vdash C}\,(\text{exchg})$$

$$\frac{\Gamma, A \vdash B \qquad \Gamma \vdash A}{\Gamma \vdash B}\,(\text{subst})$$

導出規則

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\,(\to\text{I}) \qquad \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash A}{\Gamma \vdash B}\,(\to\text{E})$$

図 1 IPC の定義

仮定から A を消してしまっても問題なさそうです。これに対応するものが (subst) で、代入 (substitution) の規則です。

以上が我々が IPC の判断に期待する構造です。ここで大事な点は (hyp) のような仮定

の使用に関する規則は体系に組込まれている一方で、(weak)(contr)(exchg)(subst) のような構造規則は「結果的に成り立ってほしい規則」である点で、実際に以下の定理が成り立ちます。

定理 1　*(weak)(contr)(exchg)(subst)* は許容規則である。すなわち *IPC* において *(weak)(contr)(exchg)(subst)* を用いた導出があった場合に、これらの規則を用いない導出が存在する。

記法 2　この記事では通常の導出規則を $\dfrac{\Gamma \vdash A}{\Gamma \vdash B}$、許容規則を $\dfrac{\Gamma \vdash A}{\Gamma \vdash B}$ と書き分ける。

以上のように、判断の構造を決める過程は二段階に分かれます。

- 判断に関する直観を反映するような規則を定める
- 体系が完成した後で期待される構造規則が許容規則になっていることを確認する

2.2. 論理結合子を導入して除去する

このように一度判断の構造が決定すると、論理結合子を定義できる段階になります。ここでは実際に「A ならば B」に対応するような論理結合子 $A \to B$ を定義しましょう。一般的に自然演繹の体系においては論理結合子の定義は導入則と除去則のペアから成ります。

導入則は「どのような状況においてその論理結合子を導入してよいか」ということを表します。$\to$ の導入則 ($\to$I) を見てみましょう。

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\,(\to\mathrm{I})$$

この導入則は「A を仮定した時に B が成り立つ」状況を $A \to B$ という命題に結びつけているわけです。

一方で除去則は「その論理結合子がある時にどのようなことを (その論理結合子を含まない形で) 結論づけられるか」を表しています。$\to$ の除去則 ($\to$E) を見てみましょう。

$$\frac{\Gamma \vdash A \to B \qquad \Gamma \vdash A}{\Gamma \vdash B}\,(\to\mathrm{E})$$

この除去則では $A \to B$ と A から B を導いています。至って妥当に見えますね。

ところで論理結合子を定義する上で導入則と除去則を雑に決めてしまってもいいのかというとそういうわけではなく、導入則と除去則が見たすべき二つの指針があります。一つ目は「期待される構造規則を壊さない」ことです。例えば以下のような $\to$ 除去の亜種を考えてみましょう。

$$\frac{\Gamma \vdash A \to B}{\Gamma, A \vdash B}\,(\to\text{E}')$$

($\to$I) の逆ですね。これでも上手くいくように思えますが、実のところこの定義だと (subst) の構造規則が許容規則でなくなってしまうためうまく行きません。このように論理結合子は判断の要求する構造の範囲内で定義されなければなりませんし、実際に ($\to$I) と ($\to$E) を体系に加えた場合には期待される構造規則は全て許容規則になります。

二つ目は「導入則と除去則が釣り合っている」ことです。先にも述べたように導入則は論理結合子を導入できる状況を、除去則は論理結合子から何を結論できるかを表しています。つまりある論理結合子の導入則と除去則は何らかの意味で対応している必要があると考えられます。哲学寄りの論理学ではこの対応を**調和** (harmony) と呼んでいて重要な研究対象となっているようですが、ここでは詳しい話は割愛します。[*2]

Pfenning と Davies の論文 [4] ではこの対応を local soundness と local completeness という二つの概念で説明しています。これらは厳密に形式化された概念ではないので、基本お気持ちベースのお話になります。

local soundness は「除去則は導入則と比較して強すぎない」という性質です。この性質は local reduction という導出の変換によって示されます。$\to$ の場合だとこれは以下のようになります。($\mathcal{D}$,$\mathcal{E}$ は何らかの導出を表すメタ変数です。)

$$\dfrac{\dfrac{\begin{array}{c}\mathcal{D}\\ \Gamma, A \vdash B\end{array}}{\Gamma \vdash A \to B}\,(\to\text{I}) \qquad \begin{array}{c}\mathcal{E}\\ \Gamma \vdash A\end{array}}{\Gamma \vdash B}\,(\to\text{E}) \;\Rightarrow_R\; \dfrac{\begin{array}{c}\mathcal{D}\\ \Gamma, A \vdash B\end{array} \qquad \begin{array}{c}\mathcal{E}\\ \Gamma \vdash A\end{array}}{\Gamma \vdash B}\,(\text{subst})$$

2　このあたりの話は [8] が参考になるのではないかと思われます

気持ちとしてはこれは「$\to$ を導入した後に除去して得た結論は、そもそも導入と除去を行わなくて導くことができる」ことを表しており、導入則が適用された時の状況以上のことを除去則が導かないことを確認できます。除去則が導入則に対して強すぎる結論を導いているとこれが成り立たなくなってしまいます。[*3]

一方で local completeness は「除去則が導入則と比較して弱すぎない」という性質です。この性質は local expansion という導出の変換によって示され、これは「除去則によって得られた判断から元の結論を復元できる」ことを表しています。$\to$ だと以下のようになり、実際に $\to$ 除去をした後に $A \to B$ を復元できることがわかります。

$$\Gamma \vdash A \to B \Rightarrow_E \dfrac{\dfrac{\dfrac{\mathcal{D}}{\Gamma \vdash A \to B}\,(\text{weak})}{\Gamma, A \vdash A \to B} \qquad \dfrac{}{\Gamma, A \vdash A}\,(\text{hyp})}{\Gamma, A \vdash B}\,(\to\text{E})}{\Gamma \vdash A \to B}\,(\to\text{I})$$

以上のように local soundness と local completeness が成り立っていることを local reduction と local expansion によってそれぞれ示すことで論理結合子の導入則と除去則が釣り合っていることを確認できます。これらの事実から $\to$ が論理結合子としてちゃんと定義できているということができます。[*4]

2.3. 論理を作る指針のまとめ

さて、ここまでのおさらいをしてみましょう。我々は以下のような手続きを踏んで直観主義論理の体系 IPC を作ったのでした。

- 判断の構造を定める
- 論理結合子を定義する

---

3 そのような例として tonk という (壊れた) 論理結合子が知られています。詳細は [8] をご覧ください。

4 Curry-Howard 同型において local reduction と local expansion は型付き λ 計算の β 簡約と η 展開に対応します [4]。論理結合子が成立していることが計算が可能であることに対応してる事実、すごく面白くないですか ⁇

- 判断の構造を壊さないように
- local soundness と local completeness を満たすように

この論理体系を作る上でまず判断の構造を定めるというやり方は新しい論理体系を作る際に便利な指針となります。次の章からは実際に IPC の構造から拡張した判断を用いて新しい論理体系をつくります。

3. 二つの階層の論理

さて、先程の IPC を土台として新しい論理を作ってみましょう。これから作るものは歴史的に Benton の随伴論理 [1] とよばれるもの亜種ですが、ここでは IPC2 と呼ぶことにしましょう。IPC2 の基本的なアイデアは、二つの階層を持たせることです。

レベル 1 の命題変数	$p, q, ...$	
レベル 1 の命題	A, B	$:= p \mid A \to B \mid \downarrow\alpha$
レベル 1 の文脈	Γ, Δ	$:= \cdot \mid \Gamma, A$
レベル 2 の命題変数	$\pi, \sigma, ...$	
レベル 2 の命題	α, β	$:= \pi \mid \alpha \Rightarrow \beta \mid \uparrow A$
レベル 2 の文脈	Θ, Σ	$:= \cdot \mid \Theta, \alpha$

定義から明らかなように、IPC2 では二つの階層で別々に記号が定義されています。レベル 2 がメタ、レベル 1 がオブジェクトの階層に対応しています。命題としては $\uparrow A$ と $\downarrow\alpha$ によって相互のレベルを行き来できるようになっています。また、二つの階層に応じて IPC2 では二つの判断が存在します。

$$\text{レベル 1 の判断} \Theta\,;\,\Gamma \vdash_1 A \qquad \text{レベル 2 の判断} \Theta \vdash_2 \alpha$$

階層の構造は非対称である点に注意してください。レベル 1 の判断ではレベル 1 と 2 の文脈がそれぞれあるのに対し、レベル 2 の判断ではレベル 2 の文脈しかありません。仮定の使用に関しては階層に応じて以下の二つのルールが定義されます。

$$\frac{A \in \Gamma}{\Theta\,;\,\Gamma \vdash_1 A}\ (\text{hyp1}) \qquad\qquad \frac{\alpha \in \Theta}{\Theta \vdash_2 \alpha}\ (\text{hyp2})$$

このように現在注目しているレベルの仮定を使用できます。 それぞれのレベルの文脈は IPC の文脈と同じようにふるまうと考えると、 以下の構造規則が成り立つことが期待されます。弱化、縮約、交換の規則も期待されますがここでは省略します。

$$\frac{\Theta\,;\,\Gamma, A \vdash_1 B \qquad \Theta\,;\,\Gamma \vdash_1 A}{\Theta\,;\,\Gamma \vdash_1 B}\,(\text{subst11})$$

$$\frac{\Theta, \alpha \vdash_2 \beta \qquad \Theta \vdash_2 \alpha}{\Theta \vdash_2 \beta}\,(\text{subst22}) \qquad \frac{\Theta, \alpha\,;\,\Gamma \vdash_1 A \qquad \Theta \vdash_2 \alpha}{\Theta\,;\,\Gamma \vdash_1 A}\,(\text{subst12})$$

このように構造が定まったので論理結合子を定めていきましょう。 まず $\to$ と $\Rightarrow$ はそれぞれのレベルにおいて IPC の $\to$ と同じように定義されます。これらの local reduction, local expansion も IPC の $\to$ とほぼ同じなので省略します。

$$\frac{\Theta\,;\,\Gamma, A \vdash_1 B}{\Theta\,;\,\Gamma \vdash_1 A \to B}\,(\to\text{I}) \qquad \frac{\Theta\,;\,\Gamma \vdash_1 A \to B \qquad \Theta\,;\,\Gamma \vdash_1 A}{\Theta\,;\,\Gamma \vdash_1 B}\,(\to\text{E})$$

$$\frac{\Theta, \alpha \vdash_2 \beta}{\Theta \vdash_2 \alpha \Rightarrow \beta}\,(\Rightarrow\text{I}) \qquad \frac{\Theta \vdash_2 \alpha \Rightarrow \beta \qquad \Theta \vdash_2 \alpha}{\Theta \vdash_2 \beta}\,(\Rightarrow\text{E})$$

$\to$ と $\Rightarrow$ が特定のレベルにおける論理結合子であったのに対して $\uparrow$ と $\downarrow$ はレベルをまたぐ IPC2 に特有の論理結合子となっています。まずは $\uparrow$ の定義から見ていきましょう。

$$\frac{\Theta\,;\,\cdot \vdash_1 A}{\Theta \vdash_2 \uparrow A}\,(\uparrow\text{I}) \qquad \frac{\Theta \vdash_2 \uparrow A}{\Theta\,;\,\Gamma \vdash_1 A}\,(\uparrow\text{E})$$

$\uparrow$ の導入規則ではレベル 2 における $\uparrow A$ の意味を「レベル 1 で A が仮定無しで成り立つこと」 と定義しています。 その帰結として除去規則ではレベル 2 で $\uparrow A$ で成り立つ時にレベル 1 で A を導きます。(なお、ここでレベル 1 の文脈に任意の Γ を許容するのはレベル 1 における弱化を成立させるためです。) 実際にこれらの規則が local reduction と local expansion をつくることを以下のように確認できます。

$$\frac{\dfrac{\dfrac{\mathcal{D}}{\Theta\,;\,\cdot \vdash_1 A}\,(\uparrow\text{I})}{\Theta \vdash_2 \uparrow A}\,(\uparrow\text{E})}{\Theta\,;\,\Gamma \vdash_1 A} \Rightarrow_R \frac{\dfrac{\mathcal{D}}{\Theta\,;\,\cdot \vdash_1 A}}{\Theta\,;\,\Gamma \vdash_1 A}\,(\text{weak1})$$

$$\begin{array}{c}\mathcal{D}\\ \Theta \vdash_2 \uparrow A\end{array} \Rightarrow_E \dfrac{\dfrac{\begin{array}{c}\mathcal{D}\\ \Theta \vdash_2 \uparrow A\end{array}}{\Theta\,;\cdot \vdash_1 A}\,(\uparrow\mathrm{E})}{\Theta \vdash_2 \uparrow A}\,(\uparrow\mathrm{I})$$

↑がレベル 2 からレベル 1 の命題を扱う論理結合子であったのに対し、↓はレベル 1 からレベル 2 の命題を扱う論理結合子です。

$$\frac{\Theta \vdash_2 \alpha}{\Theta\,;\Gamma \vdash_1 \downarrow\alpha}\,(\downarrow\mathrm{I}) \qquad\qquad \frac{\Theta\,;\Gamma \vdash_1 \downarrow\alpha \qquad \Theta,\alpha\,;\Gamma \vdash_1 A}{\Theta\,;\Gamma \vdash_1 A}\,(\downarrow\mathrm{E})$$

↓の導入規則で表されているように、$\downarrow\alpha$ は「レベル 2 において α が成り立っている」というレベル 1 の命題になっています。対応する除去則では $\downarrow\alpha$ が成り立っている時にレベル 2 の仮定に α が入っていることにして推論してもよいということを表しています。これらの規則も local reduction と local expansion を満たしていることがわかります。

$$\frac{\dfrac{\begin{array}{c}\mathcal{D}\\ \Theta \vdash_2 \alpha\end{array}}{\Theta\,;\Gamma \vdash_1 \downarrow\alpha}\,(\downarrow\mathrm{I}) \qquad \begin{array}{c}\mathcal{E}\\ \Theta,\alpha\,;\Gamma \vdash_1 B\end{array}}{\Theta\,;\Gamma \vdash_1 B}\,(\downarrow\mathrm{E}) \Rightarrow_R \frac{\begin{array}{c}\mathcal{D}\\ \Theta \vdash_2 \alpha\end{array} \qquad \begin{array}{c}\mathcal{E}\\ \Theta,\alpha\,;\Gamma \vdash_1 B\end{array}}{\Theta\,;\Gamma \vdash_1 B}\,(\mathrm{subst12})$$

$$\begin{array}{c}\mathcal{D}\\ \Theta\,;\Gamma \vdash_1 \downarrow\alpha\end{array} \Rightarrow_E \frac{\begin{array}{c}\mathcal{D}\\ \Theta\,;\Gamma \vdash_1 \downarrow\alpha\end{array} \qquad \dfrac{\dfrac{}{\Theta,\alpha \vdash_2 \alpha}\,(\mathrm{hyp2})}{\Theta,\alpha\,;\Gamma \vdash_1 \downarrow\alpha}\,(\downarrow\mathrm{I})}{\Theta\,;\Gamma \vdash_1 \downarrow\alpha}\,(\downarrow\mathrm{E})$$

さて、これで二階層の論理 IPC2 の定義が出揃いました。これで実際にどんな証明ができるか見てみましょう。

$$\frac{\dfrac{\dfrac{}{;\downarrow\uparrow A \vdash_1 \downarrow\uparrow A}\,(\mathrm{hyp1}) \qquad \dfrac{\dfrac{}{\uparrow A \vdash_2 \uparrow A}\,(\mathrm{hyp2})}{\uparrow A\,;\downarrow\uparrow A \vdash_1 A}\,(\uparrow\mathrm{E})}{;\downarrow\uparrow A \vdash_1 A}\,(\downarrow\mathrm{E})}{;\vdash_1 \downarrow\uparrow A \to A}\,(\to\mathrm{I})$$

$$\dfrac{\dfrac{}{;\downarrow\uparrow A \vdash_1 \downarrow\uparrow A}\,(\text{hyp1}) \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\uparrow A \vdash_2 \uparrow A}\,(\text{hyp2})}{\uparrow A\,;\cdot \vdash_1 \downarrow\uparrow A}\,(\downarrow\text{I})}{\uparrow A \vdash_2 \uparrow\downarrow\uparrow A}\,(\uparrow\text{I})}{\uparrow A\,;\downarrow\uparrow A \vdash_1 \downarrow\uparrow\downarrow\uparrow A}\,(\downarrow\text{I})}{;\downarrow\uparrow A \vdash_1 \downarrow\uparrow\downarrow\uparrow A}\,(\downarrow\text{E})}{;\vdash_1 \downarrow\uparrow A \to \downarrow\uparrow\downarrow\uparrow A}\,(\to\text{I})$$

以上のように $\downarrow\uparrow A \to A$ と $\downarrow\uparrow A \to \downarrow\uparrow\downarrow\uparrow A$ が成り立つことがわかります。これは様相論理における T の公理 $\Box A \to A$ と K4 の公理 $\Box A \to \Box\Box A$ に似ていますが、実際に $\downarrow\uparrow$ は S4 の様相演算子になっており[*5]、実際に Pfenning が定義している直観主義の S4 様相論理の自然演繹の体系と素直に対応をとることができます。[*6]

4. 階層の対称性と K 様相

先ほどの章では論理を二つの階層に分離し、互いの階層を相互作用できるようにすることで二階層の論理 IPC2 をつくったのでした。実はこの複数の階層のアイデアから少し進めることで様相論理をつくることができます。面白いのは IPC2 の $\downarrow$ と $\uparrow$ をそれぞれ (直観主義の)K の様相に拡張する方法があることです。

4.1. ヒルベルト流による直観主義様相論理の定式化

端的に言うと様相論理とは「命題自身の性質」を表すような論理結合子を含む論理体系のことを指します。例えば $\Box A$ は「必ず A である」、$\diamond A$ は「A の可能性がある」という意味の命題です。

様相論理は専ら古典論理の拡張としてのヒルベルト流の体系において研究がされている一方で、直観主義の体系あるいはヒルベルト流以外の体系においては比較的研究が進んでいません。そこでこの章で紹介する直観主義 K 様相論理の自然演繹の体系を紹介する前に、同じ論理のヒルベルト流による定式化 IK を紹介します。まず、次のように記号を定めます。

5　S4 様相論理がどういうものかについては次の章で軽く触れています。

6　[6] がそういう話をしていると思うけどちゃんと読んだわけではないのでなんとも言えない。

$$
\begin{array}{lll}
\text{命題変数} & p, q, \ldots & \\
\text{命題} & A, B & := p \mid A \to B \mid \Box A
\end{array}
$$

次に以下のように公理を定めます。

$$
\begin{array}{ll}
\to_1 & A \to B \to A \\
\to_2 & \big(A \to B \to C\big) \to \big(A \to B\big) \to A \to C \\
K & \Box\big(A \to B\big) \to \Box A \to \Box B
\end{array}
$$

このとき、ある命題 A が証明可能である、ということは以下のように再帰的に定義されます。

- A が公理であるとき、A は証明可能である。
- A と $A \to B$ がそれぞれ証明可能である時、B も証明可能である (Modus Ponens)
- A が証明可能である時、$\Box A$ も証明可能である (Necessitation)

これらの定義によって証明可能な命題の集合を IK とします。また、上の公理に T の公理 $\Box A \to A$ と K4 の公理 $\Box A \to \Box\Box A$ を加えることで得られる論理が直観主義の S4 様相論理 IS4 で、先の章の最後に言及したものになります。

4.2. ↓と Dual-Context Modality

一つめの拡張では↓を K 様相演算子に拡張します。この類の形式化の仕方に固定の名前があるわけではないのですが、最近の論文では Dual-Context と呼ばれています [9]。ここでは IPC2M と呼ぶことにしましょう (最後の M は Modal の M です)。

これは次の拡張にも共通することですが IPC2 の↑や↓を様相演算子に拡張する上で大事になるのは「どの階層でも同じ論理が動く」ということです。まず記号の定義を見てみてこれを確認しましょう。

命題変数	$p, q, \ldots$	
命題	A, B	$:= p \mid A \to B \mid \blacktriangledown A$
文脈	Γ, Δ	$:= \cdot \mid \Gamma, A$

IPC2M では IPC2 と違って階層によって命題の定義が分かれていません。レベル 1 でもレベル 2 でも同じ種類の命題を扱うことができます。$\blacktriangledown$ が $\downarrow$ を拡張して得られる様相演算子 (になる予定のもの) です。IPC2M では判断は以下のような形になります。

$$\Delta\,;\Gamma \vdash A$$

判断が一つしかないじゃないかと思われるかもしれませんが、これで二階層における判断をまとめています。気持ちとしては IPC2 における判断は以下のように IPC2M の判断に対応します (記号が合ってないのであくまでも気持ちです)。

$$\Theta\,;\Gamma \vdash_1 A \;\approx\; \Theta\,;\Gamma \vdash A$$
$$\Theta \vdash_2 \alpha \;\approx\; \cdot\,;\Theta \vdash \alpha$$

このように同じ形の判断で IPC2 におけるレベル 1 とレベル 2 の判断を表すようになっており、これが「どの階層でも同じ論理が動く」構造に繋っています。

基本的には IPC2M の判断が持つ構造は IPC2 のそれを踏襲したものになっています。仮定のルールはそのまんまですね。

$$\frac{A \in \Gamma}{\Delta\,;\Gamma \vdash A}\ (\text{hyp})$$

また、期待される構造規則は以下のようになっています。それぞれの文脈で弱化、縮約、交換が成立しますが場所をとるだけなので省略して代入の規則だけ書いておきましょう。

$$\frac{\Delta\,;\Gamma, A \vdash B \qquad \Delta\,;\Gamma \vdash A}{\Delta\,;\Gamma \vdash B}\ (\text{subst1}) \qquad \frac{\Delta, A\,;\Gamma \vdash B \qquad \cdot\,;\Delta \vdash A}{\Delta\,;\Gamma \vdash B}\ (\text{subst2})$$

IPC2 における (subst11) と (subst22) が (subst1) にまとめられています。(subst2) の規則も少し妙に見えるかもしれませんが、IPC2 における (subst12) を移したものだと考えるとそれらしく思えるのではないでしょうか。

これで IPC2M の判断の構造を記述したので、 論理結合子の $\to$ と $\blacktriangledown$ を定義しましょう。

$$\frac{\Delta\,;\,\Gamma, A \vdash B}{\Delta\,;\,\Gamma \vdash A \to B}\,(\to\mathrm{I}) \qquad \frac{\Delta\,;\,\Gamma \vdash A \to B \qquad \Delta\,;\,\Gamma \vdash A}{\Delta\,;\,\Gamma \vdash B}\,(\to\mathrm{E})$$

$$\frac{\cdot\,;\Gamma \vdash A}{\Gamma\,;\,\Delta \vdash \blacktriangledown A}\,(\blacktriangledown\mathrm{I}) \qquad \frac{\Delta\,;\,\Gamma \vdash \blacktriangledown A \qquad \Delta, A\,;\,\Gamma \vdash B}{\Delta\,;\,\Gamma \vdash B}\,(\blacktriangledown\mathrm{E})$$

$\to$ は IPC のそれとほぼ同じなので特に説明は必要ないでしょう。$\blacktriangledown$ の方も IPC2 の $\downarrow$ とほぼ同じですね。実際に $\blacktriangledown$ では以下のように local reduction と local expansion が成り立つので論理結合子として成り立っていることがわかります。

$$\frac{\dfrac{\overset{\mathcal{D}}{\cdot\,;\Delta \vdash A}}{\Delta\,;\,\Gamma \vdash \blacktriangledown A}\,(\blacktriangledown\mathrm{I}) \qquad \overset{\mathcal{E}}{\Delta, A\,;\,\Gamma \vdash B}}{\Delta\,;\,\Gamma \vdash B}\,(\blacktriangledown\mathrm{E}) \;\Rightarrow_R\; \frac{\overset{\mathcal{D}}{\cdot\,;\Delta \vdash A} \qquad \overset{\mathcal{E}}{\Delta, A\,;\,\Gamma \vdash B}}{\Delta\,;\,\Gamma \vdash B}\,(\text{subst2})$$

$$\overset{\mathcal{D}}{\Delta\,;\,\Gamma \vdash \blacktriangledown A} \Rightarrow_E \frac{\overset{\mathcal{D}}{\Delta\,;\,\Gamma \vdash \blacktriangledown A} \qquad \dfrac{\dfrac{}{\cdot\,;\Delta, A \vdash A}\,(\text{hyp})}{\Delta, A\,;\,\Gamma \vdash \blacktriangledown A}\,(\blacktriangledown\mathrm{I})}{\Delta\,;\,\Gamma \vdash \blacktriangledown A}\,(\blacktriangledown\mathrm{E})$$

この $\blacktriangledown$ が実際に IK における $\Box$ に一致することを示すにはリソースが足りないので、図 2 で K の公理 $\blacktriangledown(A \to B) \to \blacktriangledown A \to \blacktriangledown B$ を示す程度でお茶を濁すことにします。[*7]

4.3. $\uparrow$ と Fitch-style Modality

先ほどは 2 レベルの階層で「同じ論理を動かす」ことで $\downarrow$ が K の様相になることを確認しました。実は別の方法で「異なる階層で同じ論理が動く」ようにすることで $\uparrow$ も K の様相になります。歴史的には色々論文が出ていて Kripke-style[5] とか Fitch-style[11] とか呼ばれますが、ここでは IPCkM と呼ぶことにします。IPCkM の構造に関する基本的なアイデアは「階層の数を任意自然数個に拡張する」ことです。まずは記号の定義から。

7　証明が読みたいという人は Kavvos の arXiv 論文 [10] をあたるとよさそうです。

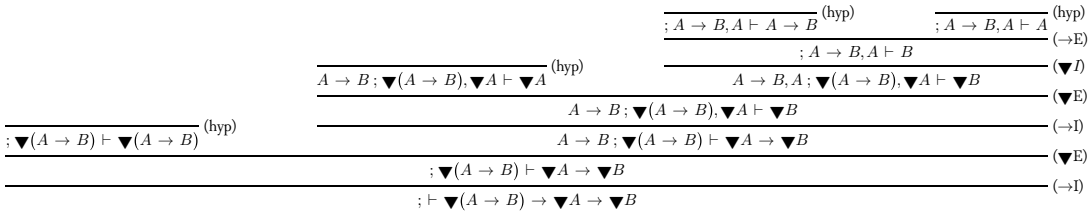


図 2 IPC2M における K の公理の証明

命題変数	$p, q, \ldots$	
命題	A, B	$:= p \mid A \to B \mid \blacktriangle A$
文脈	Γ, Δ	$:= \cdot \mid \Gamma, A$
文脈スタック	Ψ, Φ	$:= \Gamma \mid \Psi\,;\Gamma$

記法 1　以降において文脈スタックは列とみなす。例えば $\Gamma\,;\Phi$ や $\Phi\,;\Psi$ のような記法は定義上は厳密には不正だが、Γ, Φ, Ψ をそれぞれ列とみなして結合したものと見なす。

ほとんどの部分は IPC2M と同じですが、IPCkM に特徴的なのが文脈スタックです。IPC2M では階層が高々二つだったのに対して、IPCkM では任意自然数個の階層を認めるためこのような定義をして任意個の階層を表現できるようにしています。こうすることでで IPCkM の判断は以下のような形になります。

$$\Psi \vdash A$$

IPC2 では二つの階層に応じて二つの判断がありましたが、IPCkM ではどの階層でも同じ命題を扱うため判断を一つにまとめることができます。階層の数が増えても期待される構造は IPC2 と基本的に同じで、仮定のルールは以下のように表現されます。

$$\frac{A \in \Gamma}{\Psi\,;\Gamma \vdash A}\ (\text{hyp})$$

また、一部の構造規則は IPC2 のそれを素直に拡張したものとなっています。例えば代入の規則はどの階層の文脈でも成り立ち、これをまとめて以下のように表現できます。

$$\frac{\Psi\,;\Gamma, A\,;\Phi \vdash B \qquad \Psi\,;\Gamma \vdash A}{\Psi\,;\Gamma\,;\Phi \vdash B}\ (\text{subst})$$

弱化、縮約、交換もどの階層でも成り立ちますが例によって省略します。さて、IPCkM には他にも構造規則があります。IPC2 から K の様相論理への拡張の基本的な考え方は「異なる階層で同じ論理が動くようにする」と述べましたが、IPCkM ではそれが以下のような構造規則で表現できます。

$$\frac{\Psi \vdash A}{\cdot\,;\Psi \vdash A}\,(\text{levelweak}) \qquad \frac{\cdot\,;\Psi \vdash A}{\Psi \vdash A}\,(\text{levelcontr})$$

上の規則において $\cdot\,;\Psi \vdash A$ と $\Psi \vdash A$ はほとんど同じ判断に見えますが、文脈スタックの深さが違うため違う階層の判断になっています。違う階層における導出だったとしてもそれらがほぼ同じように導出できるのであればそれらを同一視しよう、というのが levelweak と levelcontr です。

さて、IPCkM の判断の構造が決まったので実際に論理結合子を定義しましょう。実のところこれらは IPC2 のそれと大差ありません。

$$\frac{\Psi\,;\Gamma, A \vdash B}{\Psi\,;\Gamma \vdash A \to B}\,(\to\text{I}) \qquad \frac{\Psi \vdash A \to B \qquad \Psi \vdash A}{\Psi \vdash B}\,(\to\text{E})$$

$$\frac{\Psi\,;\cdot \vdash A}{\Psi \vdash \blacktriangle A}\,(\blacktriangle\text{I}) \qquad \frac{\Psi \vdash \blacktriangle A}{\Psi\,;\Gamma \vdash A}\,(\blacktriangle\text{E})$$

▲ の定義は IPC2 の ↑ を拡張したものになっていて、ある階層で A が仮定無しで成り立つ場合に一つ上の階層で $\blacktriangle A$ を導入できます。以下の local reduction と local expansion からこれがちゃんとした論理結合子になっていることが確認できます。

$$\frac{\dfrac{\dfrac{\mathcal{D}}{\Psi\,;\cdot \vdash A}\,(\blacktriangle\text{I})}{\Psi \vdash \blacktriangle A}\,(\blacktriangle\text{E})}{\Psi\,;\Gamma \vdash A} \Rightarrow_R \frac{\dfrac{\mathcal{D}}{\Psi\,;\cdot \vdash A}}{\Psi\,;\Gamma \vdash A}\,(\text{weak})$$

$$\frac{\mathcal{D}}{\Psi \vdash \blacktriangle A} \Rightarrow_E \frac{\dfrac{\dfrac{\mathcal{D}}{\Psi \vdash \blacktriangle A}\,(\blacktriangle\text{E})}{\Psi\,;\cdot \vdash A}\,(\blacktriangle\text{I})}{\Psi \vdash \blacktriangle A}$$

▲ も IK の □ に一致しますが、こちらも図 3 に K の公理の証明を示すにとどめておきます。 証明を見たい方は Martini と Masini の論文 [3] などを見るといいのではないかと思います。

5. まとめと元ネタと関連する話題

この記事では Judgmental Reconstruction によって様相論理の体系を構築しました。 この際に判断の構造を徐々に拡張しくことで新しい論理体系を作りました。

- IPC
  - IPC2 (IS4)
    - IPC2M (IK)
    - IPCkM (IK)

このように異なる様相論理の論理体系が一つの枠組みの中で説明できています。 特に IPC2M と IPCkM は違う判断を持ちベースとなる論理結合子も異なるにも関わらず結果として両方とも IK と同じ証明能力を持っていることは非常に興味深い点です。 証明できる命題が同じである時、両者の違いとは何なのでしょうね。

Judgmental Reconstruction の話は概ね [4] に基いています。彼の体系では二階層の判断を定めて S4 様相演算子を定義しているというのは先にも述べた通りです。

二階層の論理の話は [2] が元ネタですが、筆者が最初にこの話題を読んだのは Pfenning の講義ノート [7] です。彼の講義ではシーケント計算をベースとして部分構造論理と様相論理と Curry–Howard 同型をいい感じに説明していて面白いのでおすすめです。

IPC2M の話は Kavvos の論文 [9] がベースになっています。 彼の論文では二階層の判断をもち K,T,K4,S4,GL のそれぞれの様相論理に対応するような 5 つの体系を提案しています。 このうち K に対応する体系がこの記事における IPC2M で、S4 に対応する体系が Pfenning の提案した体系 [4] に対応します。他の体系は今回の枠組みで扱えるのかどうかは不明ですが気になる点ではあります。

IPCkM に関してはいくつか元ネタがありますが、Martini と Masini の論文 [3] は K 様相論理に対応する体系について明確に言及しています。また、この記事では触れませんで

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\blacktriangle(A \to B), \blacktriangle A \vdash \blacktriangle(A \to B)}\,(\text{hyp})}{\blacktriangle(A \to B), \blacktriangle A\,;\cdot \vdash A \to B}\,(\blacktriangle\text{E}) \qquad \dfrac{\dfrac{}{\blacktriangle(A \to B), \blacktriangle A \vdash \blacktriangle A}\,(\text{hyp})}{\blacktriangle(A \to B), \blacktriangle A\,;\cdot \vdash A}\,(\blacktriangle\text{E})}{\blacktriangle(A \to B), \blacktriangle A\,;\cdot \vdash B}\,(\to\text{E})}{\blacktriangle(A \to B), \blacktriangle A \vdash \blacktriangle B}\,(\blacktriangle\text{I})}{\blacktriangle(A \to B) \vdash \blacktriangle A \to \blacktriangle B}\,(\to\text{I})}{\vdash \blacktriangle(A \to B) \to \blacktriangle A \to \blacktriangle B}\,(\to\text{I})$$

図 3 IPCkM における K の公理の証明

したが IPCkM の判断の構造に更に拡張を加えることで T,K4,S4 の様相論理を得ることができます。これについては Davies と Pfenning[5] の論文で言及されています。

参考文献

[1] P. N. Benton. A Mixed Linear and Non-Linear Logic: Proofs, Terms and Models (Extended Abstract). In *Computer Science Logic*, pages 121–135, 1995.

[2] Per Martin-Löf. On the Meanings of the Logical Constants and the Justifications of the Logical Laws. *Nordic Journal of Philosophical Logic*, 1(1), pages 11–60, 1996.

[3] Simone Martini and Andrea Masini. A Computational Interpretation of Modal Proofs. *Proof Theory of Modal Logic*, 2, pages 213–241, 1996.

[4] Frank Pfenning, Rowan Davies. A judgmental reconstruction of modal logic.. *Mathematical Structures in Computer Science*, 11(4), pages 511–540, 2001.

[5] Rowan Davies and Frank Pfenning. A Modal analysis of Staged Computation. *Journal of the ACM*, 48(3), pages 555–604, 2001.

[6] Json Reed. *A Judgmental Deconstruction of Modal Logic*. `http://www.cs.cmu.edu/~jcreed/papers/jdml.pdf`, 2009.

[7] Frank Pfenning. *Lecture Note on Of Cource!*. `https://www.cs.cmu.edu/~fp/courses/15816-f16/lectures/15-ofcourse.pdf`, 2016.

[8] Shunsuke Yatabe. 証明論的意味論入門 . `http://www.academia.edu/31173006/`

証明論的意味論入門 , 2017.

[9] G. A. Kavvos. Dual-Context Calculi for Modal Logic. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12, 2017.

[10] G. A. Kavvos. *Dual-Context Calculi for Modal Logic*. `https://arxiv.org/abs/1602.04860`, 2018.

[11] Ranald Clouston. Fitch-Style Modal Lambda Calculi. In *21st International Conference on Foundations of Software Science and Computation Structures (FoSSaCS)*, pages 258–275, 2018.

あとがき：SATYSF$_{\mathrm{I}}$ で技術系同人誌を書いた話

このたびは yabaitech.tokyo vol.1 をご購入いただきありがとうございます。この同人誌は大学院の同期で何か同人誌書いてみないかというユルいノリで始まったもので、途中には様々な犠牲が発生しました[*8]がなんにせよ出版までこぎつけられてよかったです。実はこの同人誌は gfn 氏の開発する組版システム SATYSF$_{\mathrm{I}}$ を用いて組版されています。あとがきではそのあたりの話をしようかと。

今回 SATYSF$_{\mathrm{I}}$ を採用した理由としては gfn が yabaitech.tokyo に参加したことが大きいです[*9]。何か問題に直面した時でも直接開発者に質問できるというのはかなり恵まれた環境です。実際そのおかげでここまで来ることができました。

SATYSF$_{\mathrm{I}}$ の組版処理システムとしての能力はこの同人誌自体が証拠になっています。組版がちゃんとしているのはもちろん、目次の自動生成や図、脚注、相互参照、参考文献リストなど本を作るのに必要な機能を一通り扱うことができます。数式関連機能も充実しており、証明図を組むことができます。SATYSF$_{\mathrm{I}}$ の最大の特徴は ML 風の型システムと構文を言語に取り入れていることですが、実際に使ってみるとその有用さが実感できます。エラーが起きたとしてもどの部分がどのように悪いかをしっかり指摘してくれますし、種々のマクロについて型さえわかってしまえばどのようにそれを用いればいいかわかるのはかなり便利です。内部のインターフェースも型のおかげでどのような仕組

8 nullpo_head は内臓を痛め、wasabiz は自転車で転倒し手を骨折。どうしてこうなった

9 彼はこの合同誌には寄稿してはいませんが、yabaitech.tokyo 名義で The SATYSF$_{\mathrm{I}}$ Book を頒布しています。

みで動いているかがなんとなく把握できるため独自のマクロを作ることも (tex 言語に比べれば) 容易です。関数型の言語で組版をするのは新鮮で楽しい体験でした。

一方で周辺ツールやライブラリに関してはまだ発展途上です。例えば同人誌に用いたクラスファイルは gfn が The SATYSFI Book を作成するのに用いたものをベースに改造して作っています[*10]。それなしで独自に同人誌を作るのは厳しかったでしょう。インストールプロセスの改善、パッケージマネージャ、bibtex-like な書誌情報の管理、豊富な記号等々、欲しいけれども実装されていないものはたくさんあります。しかしながらこれはチャンスでもあります。今回 The SATYSFI Book が出版されたので SATYSFI への情報のアクセスは大分容易になるはずです。開発者のみなさん、周辺ツールやライブラリを作って SATYSFI にコントリビュートするなら今です！

10 各記事に著者 / 参考文献リストを書いたり目次の見栄えを変えたり。このあたりは MasWag が率先してやってくれました。

yabaitech.tokyo vol.1

発行日	2018.10.08
サークル	yabaitech.tokyo
Web サイト	http://yabaitech.tokyo
連絡先	admin@yabaitech.tokyo
印刷所	株式会社ポプルス