第 7 講 決定的有限状態機械

林　恒俊

有限状態機械

- FSA は構成が最も限定されている抽象機械のクラスである。すなわち FSA は制御部分と入力テープのみで構成される。
- FSA の状態遷移を決定する入力は現状態とヘッドが参照するテープ上の記号の 2 個しかない。
- FSA の入力テープは 1 ステップ毎に自動的に前進するように定義されるので状態遷移の出力は制御部の次状態だけである。テープ操作は行わない。
- FSA の状態遷移は現状態と入力記号をパラメータ、次状態を値とする関数ないしは関係でありこれらの 3 要素の組の集合として定義される。
- ある入力パラメータ (引数) に対して次状態がユニーク (一意的) である (状態遷移関数となる) 時の FSA を**決定的有限状態機械** (**deterministic finite state automaton**, **DFA**) という。
- ある入力パラメータについて状態遷移表が複数個の次状態を持つ (状態遷移関係となる) 時、あるいは入力記号を読まないで次状態をとるような FSA を**非決定的有限状態機械** (**nondeterministic finite automaton**, **NFA**) という。これは遷移先が状態の集合であると見なすことができる。

DFA の形式的定義

- DFA は 5 つ組 $(Q, \Sigma, \delta, q_0, F)$ により形式的に定義する。すなわち
  - Q は状態の有限集合

- Σ はアルファベット
- δ は $\delta : (q, \sigma) \to q'$ または $q' = \delta(q, \sigma)$ で定義される状態遷移関数、ただし $q, q' \in Q, \sigma \in \Sigma$
- q_0 は初期状態、ただし $q_0 \in Q$
- F は終了状態集合、ただし $F \subseteq Q$

の 5 つの要素を与えて定義する。

- 状態遷移関数 δ の領域は状態と記号の組であり、値域は状態である。すなわち δ の引数は現在の状態と現在ヘッドが読んでいる記号で、値は次状態と解釈される。
- 状態遷移関数は 3 つ組の集合 $\delta = \{(q, \sigma, q') \mid q, q' \in Q, \sigma \in \Sigma\}$ と表示されることもある。
- DFA の状態遷移関数ではある引数について次状態は高々1 個までである。すなわち現状態と入力記号を決めれば次状態が 1 通りだけ決定するか次状態が定義されないかのどちらかである。
- 状態遷移関数を表に表現したものが状態遷移表である。通常表の上下方向に現状態、左右方向に入力記号、表の要素に次状態を記入する。

DFA の実例

- DFA M_0 を次のように定義する。

$$M_0 = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_0\})$$
$$\text{ただし} \quad \delta = \{(q_0, 0, q_0), (q_0, 1, q_1), (q_1, 0, q_1), (q_1, 1, q_0)\}$$

- M_0 の状態集合は $\{q_0, q_1\}$、アルファベットは $\{0, 1\}$、初期状態は q_0 で終了状態集合は $\{q_0\}$ である。
- 状態遷移関数 δ は現状態を行、入力記号を列にもつ状態遷移表とし

ても表現される。

		入力記号	
		0	1
状態	q_0	q_0	q_1
	q_1	q_1	q_0

DFA の動作と言語の定義

- DFA の動作は非形式的に概ね次のように述べられる。
- DFA は与えられた入力記号列について動作する。記号列はあらかじめ入力テープ上に記入されている。動作開始時にヘッドが記号列の先頭に位置づけられる。
- DFA は初期状態から動作を開始する。
- DFA は次の 1. から 2. を可能な限り停止せずに繰返す。
  1. 現状態と現入力記号 (ヘッドが参照する記号) から状態遷移関数に基づいて次状態を決定する
  2. テープを進めると同時に次状態に移行する
- 以下のように動作を継続できない場合 DFA は停止する。
  - テープ上の記号列をすべて読込むか
  - 次状態が定義されていない
- DFA が停止した時に
  - 入力記号列をすべて読込みずみでかつ
  - 状態が終了状態集合の要素である場合

  その DFA は入力記号列を受理したという。
- DFA が入力記号列を受理しない場合、その DFA は入力記号列を拒否したという。
- DFA が定義する言語とはその DFA が受理する記号列の集合である。

DFA の動作例

- 入力記号列 0110 に対して DFA M_0 は次のような段階を追って動作する。

現状態	入力	遷移関数	次状態	備　考
q_0	0	$(q_0, 0) \to q_0$	q_0	初期状態
q_0	1	$(q_0, 1) \to q_1$	q_1	
q_1	1	$(q_1, 1) \to q_0$	q_0	
q_0	0	$(q_0, 0) \to q_0$	q_0	
q_0				停止

  M_0 は停止した時の状態 $q_0 \in F$ であるため記号列 0110 を受理する。

- 入力記号列 010 に対して DFA M_0 は次のような段階を追って動作する。

現状態	入力	遷移関数	次状態	備　考
q_0	0	$(q_0, 0) \to q_0$	q_0	初期状態
q_0	1	$(q_0, 1) \to q_1$	q_1	
q_1	0	$(q_1, 0) \to q_1$	q_1	
q_1				停止

  M_0 は停止した時の状態 $q_1 \notin F$ であるため記号列 010 を拒否する。

DFA の動作の形式的定義

- DFA の動作を動作状態を表す表現式とその動作状態の段階を与える演算子により形式的に定義する。
- DFA の動作状態は DFA の状態とテープの状態を与えれば定義できる。したがってこの 2 要素の組で DFA の**動作状態** (**machine configuration**) を定義する。DFA $M = (Q, \Sigma, \delta, q_0, F)$ の動作状態を $\mathcal{C}$ とすると

$$\mathcal{C} = (q, [p, x])$$

  で与えられる。ただし $q \in Q$ で $[p, x]$ はテープ構成である。

- この式は DFA の現状態が q でかつ記号列 x の p 番目の記号を読んでいることを表示する。なお M の動作状態の集合を $C(M)$ で示す。

$$C(M) = \{(q, [p, x]) \mid q \in Q, x \in \Sigma^*, 1 \leq p \leq |x| + 1\}$$

- 入力記号列が x なら DFA の初期動作状態 $\mathcal{C}_0 \in C(M)$ は

$$\mathcal{C}_0 = (q_0, [1, x])$$

と表現される。

- DFA の動作状態の 1 段階進行を表す関係演算子を $\vdash_M$ と表示する。この関係は DFA に依存するので M が添えられている。

- $\mathcal{C}_1 = (q_1, [p_1, x])$ 及び $\mathcal{C}_2 = (q_2, [p_2, x])$ を M の任意の 2 個の動作状態として $(\mathcal{C}_1, \mathcal{C}_2 \in C(M))$、次の 2 条件が成立すれば

$$\begin{cases} p_1 + 1 = p_2 & (\text{ヘッドが 1 記号分前進している}) \\ \delta(q_1, x(p_1)) = q_2 & (\text{次状態が遷移関数の値と同じ}) \end{cases}$$

$\mathcal{C}_1 \vdash_M \mathcal{C}_2$ と表示し DFA M は動作状態 $\mathcal{C}_1$ から動作状態 $\mathcal{C}_2$ に**進む** (**next move**, **step**, **yield**) という。

- すなわち $\mathcal{C}_1 \vdash_M \mathcal{C}_2$ は DFA の 1 段階動作の状態変化の関係を示している。$\vdash_M$ は DFA 動作状態間で定義される関係演算子であり $\vdash_M \subseteq C(M) \times C(M)$ である。

- DFA は次状態が定義されていない時停止する。いいかえると状態遷移関数が未定義である時に停止する。DFA の**停止動作状態** (**halting configuration**) は $\mathcal{C}_h = (q, [p, x])$ で次状態 $\delta(q, x(p))$ が未定義の場合である。

- ヘッドが記号列をすべて読み、テープの最後に到達した場合状態遷移関数が未定義になるため DFA は停止する。この動作状態を**最終動作状態** (**final configuration**) といい

$$\mathcal{C}_h = (q, [|x| + 1, x])$$

である。

DFA 動作状態例

- 上記の DFA M_0 において入力記号列を 0110 とすると

$$\begin{aligned}(q_0,[1,0110]) &\vdash_{M_0} (q_0,[2,0110])\\ (q_0,[2,0110]) &\vdash_{M_0} (q_1,[3,0110])\\ (q_1,[3,0110]) &\vdash_{M_0} (q_0,[4,0110])\\ (q_0,[4,0110]) &\vdash_{M_0} (q_0,[5,0110])\end{aligned}$$

  で動作のすべての段階が与えられる。

- 初期動作状態は $(q_0,[1,0110])$ で最終動作状態は $(q_0,[5,0110])$ である。

言語定義

- M を動作関係演算$\vdash_M$ が定義された DFA とする。
- $\mathcal{C}_i \in C(M)$ ただし $0 \leq i \leq n$ を $n+1$ 個の一連の動作状態とする。
- もし一連の式

$$\mathcal{C}_0 \vdash_M \mathcal{C}_1, \mathcal{C}_1 \vdash_M \mathcal{C}_2, \ldots, \mathcal{C}_{n-1} \vdash_M \mathcal{C}_n$$

  が成立するとき

$$\mathcal{C}_0 \vdash_M^* \mathcal{C}_n$$

  と表示する。$\vdash_M^*$ は継続機械動作を代表する関係演算を示している。これを次のように表示することもある。

$$\mathcal{C}_0 \vdash_M \mathcal{C}_1 \vdash_M \mathcal{C}_2 \vdash_M \cdots \vdash_M \mathcal{C}_n$$

- この式は DFA M による**計算過程** (**computation**, **trace**) を示すものと考えられる。
- 上記のの例は $(q_0,[1,0110]) \vdash_{M_0}^* (q_0,[5,0110])$ と書くことができる。
- DFA M で入力 x について次の条件が成立する時 M は記号列 x を受理するという。

$$(q_0,[1,x]) \vdash_M^* (f,[|x+1|,x]) \quad \text{かつ} \quad f \in F$$

  すなわち

- DFA の初期動作状態から計算を開始し
- 入力をすべて読みとった後で停止し
- その時の状態が終了状態のいずれか

である。

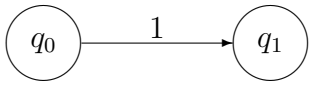
- $(q_0, [1, x]) \vdash_M^* (f, [|x+1|, x])$ は $(q_0, \tau^{Initial}(x)) \vdash_M^* (f, \tau^{Final}(x))$ とも表示される。
- 入力 x について DFA M が受理しない場合 M は記号列 x を拒否したという。
- DFA M が定義する言語を $L(M)$ とすると $L(M)$ は次のように記述される。

$$L(M) = \{x \mid M \text{ は } x \text{ を受理する } \}$$

- 上記の DFA M_0 は記号列 0110 を受理する。なお M_0 が受理する言語は偶数個の 1 を含む記号列からなる言語である。(確かめよ)

状態遷移図

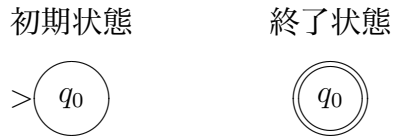
- FSA は状態を節点、記号を枝とするグラフにより表示することができる。ある状態から次状態への遷移を枝で結合した節点で表現する。グラフ形式で表示した FSA の定義を**状態遷移図** (**state transition diagram**) と呼ぶ。
- 節点は状態を表示する記号を丸で囲んで示す。枝は遷移方向を示すために矢印が使われる。例えば遷移関数に $(q_0, 1, q_1)$ という要素があると



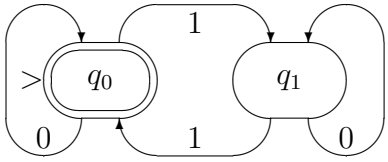
と表示される。

- 初期状態及び終了状態はそれを示す脚色をすることで示される。初

期状態は内向きの矢尻、終了状態は 2 重丸が使われることが多い。



- 状態遷移図による表示は動作を比較的直感的に理解しやすいため FSA を設計する場合によく利用される。
- 上記の M_0 は次のような状態遷移図で表現される。



考察

構文図 (**syntax diagram**) はプログラミング言語の構文定義に使われている。DFA の状態遷移図と比較すれば互いによく似ていることに気がつくだろう。

構文図と状態遷移図は双対的な関係にある。構文図の終端記号や非終端記号のノードが状態遷移図の矢印上のラベルに対応し、構文図の矢印が状態遷移図の状態ノードに対応する。構文図では状態には名前がつけられていないが DFA が受理する言語は状態遷移図の状態名に影響を受けない。矢印のラベルの記号と接続状態が言語を決定する。

構文図では非終端記号を矢印のラベルにすることが可能であり、非終端記号のラベルはそのラベルの図に移動することを示している。この移動は再帰的に行われるので、構文図を **RTN** (**recursive transition network**) と呼ぶことがある。