6. モノイド

2023 秋期「哲学者のための数学」授業資料（大塚淳）

ver. 2023 年 11 月 2 日

1　モノイドとは何か，なぜそれを学ぶのか

前章で見た位相は，空間に関する幾何学的な概念であった．一方本章の主題となるモノイドや群は，本質的に代数的な「計算」にまつわる概念である．よって我々は，代数，幾何と来て我々は再び代数の世界に戻ってきた．

文系の学生にとって，「位相」という言葉は耳にしたことくらいはあっても，「モノイド」や「群」となると聞いたこともない，という人も多いかもしれない．しかし実のところ我々は皆，小学生のころからモノイドや群に親しんでいる．というのも，足し算や掛け算などはまさにこのモノイドや群の作用に他ならないからだ．モノイドや群は，そうした四則演算を始めとした「演算」一般の最もプリミティブな形を抜き出したものと言える．それ以外にも，モノイドは対象や系の変化・発展を表すために用いられるし，また群はモノの対称性（symmetry）の数学的な表現を与える．そしてこの対称性という考え方は，物理学における「法則性」という考えを裏から支えるものであり，また哲学的には客観性の概念と深い結び付きを持っている．こうしたことから，モノイドや群は非常に広範な科学的・哲学的含意を有している．

現代物理学を始め様々な科学で応用される群論は，極めて高度に発展しており，その全体像を掴むことを容易ではない．しかしその基本的な考え方はこれ以上ないくらいシンプルである．ここではその本質的な点のみに的を絞って紹介したい．そこから得られるモノイドや群は，数学者や物理学者からしたらおもちゃみたいに簡単なものでしかないかもしれないが，その哲学的含意を考えるには十分であろう．

2　モノイド

まずはいつものように，集合をベースにモノイドを定義しよう．

定義 2.1: モノイド

合 M 上に，積と呼ばれる二項写像 $\circ : M \times M \to M$ が定義されており，以下の条件を満たすとき，組 $(M, \circ, i)$ を**モノイド** (monoid) という．

1. M の任意の元 l, m, n に対して，結合律 $(l \circ m) \circ n = l \circ (m \circ n)$ がなりたつ．
2. **単位元** (identity element) と呼ばれる元 $i \in M$ が存在して，M の任意の元 m に対して，$i \circ m = m \circ i = m$ がなりたつ．

これだけである．つまりモノイドとは，その 2 つの元 m, n をある元 $m \circ n$ に対応させるモノイド演算 $\circ$ が備わっているような集合である．公理 1 は，この演算が結合律を満たすこと，そして公理 2 はこの演算において「何もしない」単位元が存在することを言っている．しばしば演算記号は省略され，$m \circ n$ は mn のように書かれる．また誤解が生じないときは，演算や単位元を明示せずに単に M がモノイドである，というように言うこともある．

事例 2.1

ゼロを含む自然数 $\mathbb{N}$（つまり非負整数）は，二項演算 $+$ とモノイドをなす．ここでの単位元は 0 である．実際任意の自然数 x, y, z について，$(x + y) + z = x + (y + z)$ かつ $0 + x = x + 0 = x$. 同様に，$\mathbb{N}$ が乗算 $\times$ についてもモノイドとなることを確認せよ (その単位元はなんだろうか)．

事例 2.2

足し算についての上の議論は，自然数の代わりに有理数 $\mathbb{Q}$，実数 $\mathbb{R}$ のゼロ以上の部分を用いても成立する．例えば $\mathbb{R}^+ := \{x \in \mathbb{R} | x \geq 0\}$ と定義すると，$(\mathbb{R}^+, +, 0)$ はモノイドである．(負の部分はどうなるのか，と思うかもしれないが，これはあとで群を定義するときに見る．)

モノイドの演算が具体的にどうなっているのかは，それぞれの元のペアの演算結果を明示することによって表示できる．例えば，お馴染みの「九九の表」は，自然数の掛け算モノイドの演算を表で表したものだ：

	1	2	3	…	n	…
1	1	2	3	…	n	…
2	2	4	6	…	$2n$	…
3	3	6	9	…	$3n$	…
⋮	⋮	⋮	⋮		⋮	
m	m	$m2$	$m3$	⋯	mn	⋯
⋮	⋮	⋮	⋮		⋮	

九九表の各マスは，モノイド $(\mathbb{N}, \times, 1)$ の 1 から 9 までの各元（一番左の列）が，それぞれ 1 から 9 までの元（一番上の行）をどの自然数に対応させるかを表している．すべてのモノイド演算は，原理的にこうした表（「積表」という）によって表すことができる．つまり我々は小学生のころからモノイドを知っていたのである！

発展 2.1

我々は今まで，(非負) 実数 $\mathbb{R}$ を様々な数学的構造として見てきた．まず *2* 章ではそれが非

可算無限集合であることを確認した．*3* 章ではその要素の間に大小関係 $\leq, \geq$ を入れた全順序集合として見た．*4* 章では，実数を開区間 (a, b) からなる開集合を持つ位相空間として特徴づけた．そしてここでは，二項演算 $+$ および $\times$ が定義されたモノイドとして定義した．このように，同じ「実数の集合」でも様々な顔を持ち，それらの顔はすでに見たような公理によって構成される．我々が普段何気なく使う実数は，実はこうした顔全てをあわせもつ存在なのである．

もちろん，実数の特徴づけはこれで終わりなわけではない．まず引き算と割り算の導入がまだであるし（これは以下で群のところで見る），またここで導入した足し算と掛け算が互いにどう関係し合うのか（例えば分配法則 $a(b+c) = ab + ac$ が満たされるか）などは，別個の公理によって定めなければならない．このためにはさらに環 *(ring)*，体 *(field)* といった概念を導入しなければならいのだが，本授業ではそこまでは扱わない．

モノイドの例は数学以外にも事欠かない．対象を繋げたり，行為を続けて行ったりというような逐次的なプロセスは，自然にモノイド構造を生み出す．

事例 2.3: 自由モノイド

3 つの記号 a, b, c に空文字 i を加えた集合を $\Sigma = \{a, b, c, i\}$ とする．これらの文字（列）を横に繋げたものを積で表す（e.g. $a \circ b = ab, aabc \circ bac = aabcbac$）．$i$ は空文字なので繋げても何も変化しないとする（e.g. $ai = ia = a$）．積による合成を再帰的に繰り返すことで，無限に文字列を生成することができる．このように生成された文字列の集合を Σ_* とすると，これは $\circ$ を積，i を単位元としたモノイドになっており，これを Σ 上に生成される**自由モノイド** (free monoid) という．「消去 (backspace)」を使わずにパソコンのキーストロークによって得られる文は，こうした自由モノイドの一例である．

この例では，上の自然数の例とは異なり，有限個の「素材」Σ を元手として，それを積で合成することで Σ_* における無数のモノイド元を生成している．生成されたモノイドが自由であるとは，それらの元の間に自明でないような等号関係がない，ということを意味している．実際，上の Σ_* においては，いかなる文字列もそれ自身以外とは等しくない．一方自然数のほうは，例えば $7 + 5 = 2 + 10 = 12$ のように，複数の元の間に等号関係がある．

事例 2.4: ロボットのプログラム

プログラムによって操作できるロボットを考える．ロボットに可能な動きは一歩前進する (a)・右を向く (b)・左を向く (c)・何もしない (i) の 4 つとする．プログラムはこれら可能な 4 つの動作を有限回繰り返したものとして書かれる．例えば $aaba$ を入力すると，ロボットは前に二歩進んだあと右に一歩進む．任意のプログラムは逐次的につなげることができる．例えば $aaba \circ cica$ は $cica$ を実行した後に $aaba$ を実行する．このロボットのプログラムは，上と同じようにモノイド Σ_* を生成する．

これは前の事例 2.3 と構造上ほぼ同じように見えるが，重要な違いがある．事例 2.3 の記号連結は自由モノイドを生成した．一方，ロボットのプログラムの方はどうであろうか．確かに，$bcaa$ は前に二歩進んで左・右，$aabc$ は左・右を向いてから前に二歩，とロボットの動作としては異なる．その意味において，ロボットの**動作**のプログラムとしてはこちらも自由モノ

イドである．しかし，プログラムの最終的な**結果**にだけ着目すると，両者とも同じ，単に前に二歩進む (aa) のと同じ結果をもたらす．よって結果の観点からは $bcaa = aabc = aa$ というような等号関係が成立する．その意味では，このモノイドには単に動作を組み合わせてできる様々なプログラムが含まれているだけでなく，それらのプログラムの間の等号関係を含んでいる．よってここでの Σ_* は基本動作の集合から生成されたモノイドではあるが，自由に生成されたものではない．

> **事例 2.5: 人生**
>
> 我々の人生は選択の連続である．毎日の行動の選択肢を (a) 遊ぶ，(b) 勉強する，(c) バイトに行く，(d) 休むとし，これに名目的な単位元 (i) を加えたものを M とする．行動 m をした翌日に n を行うことを積 $n \circ m$ で表すと，M はモノイドとなる．例えば $adcabba$ は遊び・勉強・勉強・遊び・バイト・休み・遊びの順で過ごした一週間を表している．人生は上述の選択肢 M から生成されるモノイドである．

この事例は，人生が日々の限られた選択から生成されていく様子をモデル化している．これは自由モノイドだろうか？自由モノイドであれば，事例 2.3 で一度繋げた文字が取り消せなかったように，一回した選択は取り消しができない，つまり過去を変えることはできない．一方，自由モノイドでなければ，過去の挽回が可能である．例えば $bba = b$ という関係性（これは上のモノイドとは別の公理として導入されることになる）は，一日遊んだ後に二日続けて勉強すればそれを帳消しにできる，ということを述べている．

ここではさしあたり，人生とは自由モノイドであると仮定しよう．今，月曜から木曜までを $daaa$ と遊び呆けて過ごしたとする．このとき，どのように残りを過ごしても，月曜日に遊びに行った事実は変えられない，つまりここから $daab$ という「人生」には到達できない．ここから，自由モノイドには「元 m は元 n から生成可能である」という半順序 $n \preceq m$ が定義できることがわかる（これが半順序であることを確認せよ）．この半順序は図 1 のような木構造をなす（ここではスペースの制約のため「遊ぶ」と「勉強」という二つの選択肢のみを描いている）．この木構造は，自由モノイドによって生成される可能性の一つ一つを，根本から葉先まで続く一本の枝として表している．

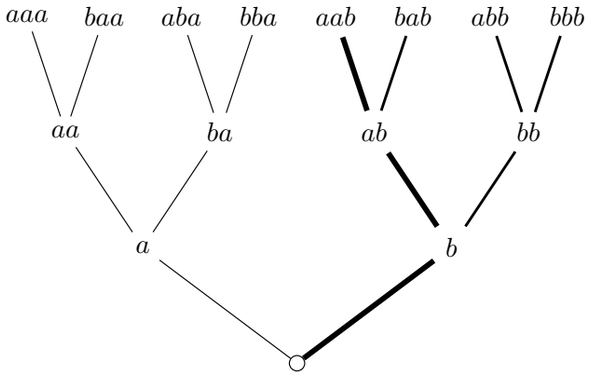

図 1　人生モノイドに対応する半順序．ここではスペースの制約のため，選択 a, b のみを考慮し，3 日間のみの分岐を表している．それぞれの枝は異なった「3 日分の過ごし方」を表している．太線は遊び・勉強・勉強と過ごした 3 日間である．2 日目の ab の時点で，他の 6 つの可能性は閉ざされている．

練習問題 2.1

上では人生を自由モノイドとしてしまったが，それは本当にそうなのだろうか．自由モノイドでない人生モデルは，人生をどのようなものとして捉えることになるだろうか．またそのとき，上で定義した生成可能関係は半順序をなすだろうか．

3 可換性

上で見てきたようにモノイドには様々なものがあるが，その中で一番重要な区分は，それが可換であるか否か，というものである．モノイド M が**可換** (commutative) であるとは，その積の結果が順序によらないこと，つまり任意の 2 つのモノイド元 $m, n \in M$ について $m \circ n = n \circ m$ がなりたつことをいう．そうならないものが一例でもあるときは，非可換 (noncommutative) であるという．

事例 3.1

足し算や掛け算は可換性が満たされる典型例である（任意の数につき $m + n = n + m, m \cdot n = n \cdot m$）．一方，$n \times n$ 行列の集合は行列積と単位行列 I によりモノイドをなすが，これは可換性を満たさない（2 つの行列 A, B について一般に $AB \neq BA$）．

足し算や掛け算に慣れ親しんだ身には，可換性は極めて一般的な性質に映るかもしれない．しかし上述の定義の通り，それは非常に強い性質である．モノイドとして表されるような現実世界の「作用」に目を向けると，多くの場面において可換性は必ずしも成立しない．例えば上の事例で見た自由モノイドによる記号列の生成（2.3）やロボットのプログラム（2.4）は明らかに非可換である．非可換性を証明するためには，モノイド元のうち，$m \circ n \neq n \circ m$ であるようなものを一組でも見つければ良い．

事例 3.2

ある人の心に生じる様々な心的刺激の集合を M で表すと，これはモノイドを生成する．その元は特定の質を持つ心的刺激，例えば「痛み (m)」や「熱さ (n)」であり，単位元は無刺激，積による合成 $m \circ n$ は熱さの後に痛みを感じることである．このとき，M は明らかに可換ではないだろう．というのも，痛みを感じてから熱さ感じることと，その逆とは明らかに別の感覚刺激だからだ．

練習問題 3.1

事例 2.5 の人生モノイドは可換だろうか，非可換だろうか．理由も含めて考えよ．

練習問題 3.2

全体として非可換なモノイドであっても，特定の元の積では可換性が満たされうる．事例 2.4 において，その実行結果（動作ではなく）が (i) 非可換，(ii) 可換となるようなプログラム元のペアをそれぞれ挙げよ．

練習問題 3.3

数学以外の事例で，可換／非可換モノイドによってモデル化できそうな現象をそれぞれ一つ挙げよ．その場合の合成と単位元がそれぞれ何に相当するのかを明示すること．

ところで我々はすでに 4 章で，束の準同型写像を写像と演算（結び・交わり）可換性によって特徴づけたのだった．そこでの可換性と，モノイドの可換性，関係はあるのだろうか？もちろんある．可換モノイドとは，以下の可換図式がすべての元 $m, n \in M$ に対して成立することにほかならない：

$$\begin{array}{ccc} M & \xrightarrow{m} & M \\ {\scriptstyle n}\downarrow & & \downarrow{\scriptstyle n} \\ M & \xrightarrow{m} & M \end{array}$$

つまり任意の M の元に対して，時計回りに $n \circ m$ と合成しても，反時計回りに $m \circ n$ と合成しても結果は同じ，ということだ．

4 準同型写像

本講ではこれまで，何らかの数学的構造を定義した際には，必ずその構造同士を対応させる，ないしその構造を保存する写像を考えてきた（e.g. 束準同型，連続写像等）．同じように，モノイド間の構造を保つ写像を考えよう．実際，我々は上で，モノイドの例として足し算と掛け算があることを見たが，これは別の言い方をすれば，足し算と掛け算はモノイドとして見れば同じ構造を持つ，ということである．両モノイドを橋渡しする関係性，つまり足し算を掛け算へとシステマティックに変換するルールが，モノイド間の準同型写像である．

定義 4.1: モノイド準同型

2 つのモノイド $(M, \circ, i), (M', \circ', i')$ が与えられているとき，写像 $f : M \to M'$ で，任意の $m, n \in M$ について次を満たすものを，M と M' の間の**準同型写像** (homomorphism) という：

1. モノイド演算を保存する: $\quad f(m \circ n) = f(m) \circ' f(n)$
2. 単位元を単位元に送る: $\quad f(i) = i'$

束準同型と同様に，この定義はモノイド間写像 f がモノイド演算 $\circ$ と可換であることを要求している：

$$\begin{array}{ccc} m, n & \xrightarrow{f} & f(m), f(n) \\ {\scriptstyle \circ}\downarrow & & \downarrow{\scriptstyle \circ'} \\ m \circ n & \xrightarrow{f} & \begin{array}{l} f(m) \circ f(n) \\ = f(m \circ' n) \end{array} \end{array}$$

つまり M 上で積を計算して f で M' へと飛ばしても（反時計回り），まず M' に飛ばしてから M' 上の積 $\circ'$ を計算しても，結果は同じということだ．モノイドの構造はその演算のあり方によって決定されるのだから，このように作られた準同型写像はモノイドの構造をしっかり保っているといえる．

ではゼロ以上の実数 $\mathbb{R}^+ := \{x \in \mathbb{R} | x \geq 0\}$ 上の足し算と掛け算のモノイドを例にとり，実際にこのような写像を構成してみよう．つまり $M := (\mathbb{R}^+, +, 0), M' := (\mathbb{R}^+, \times, 1)$ としたときの準同型 $f : M \to M'$ を見つけたい．それは無数にあるのだが，一つの例として関数

$$2^{()} :: m \mapsto 2^m$$

を考えてみよう．まずこれは $m \in \mathbb{R}^+$ から $2^m \in \mathbb{R}^+$ への関数になっている．さらに

$$2^{m+n} = 2^m \cdot 2^n$$

なので，和を積へとしっかりと移している．これを「2 を底とする指数関数」という．2 に限らず，$a > 0$ を底とする指数関数はすべて足し算としての非負実数から掛け算としての非負実数への準同型を与える．特に底としてネイピア数 $e \approx 2.718$ をとるものを単に指数関数 (exponential function) とよび，exp() と表す．

事例 4.1

我々は上で，自然数 $\mathbb{N}$ 上の足し算と非負実数 $\mathbb{R}^+$ 上の足し算はともにモノイドであると述べた．これらの間にも準同型がある．いま埋め込み $i : \mathbb{N} \to \mathbb{R}^+$ を，$i(m) = m$ で定義する．つまり i はある数 m をとって同じ数 m を返す．ただしここで入力される数 m は整数であるが（つまり $m \in \mathbb{N}$），出力される数 $i(m)$ は実数として解釈されている（つまり $i(m) \in \mathbb{R}^+$）点に注意しよう．当然 $i(m + n) = i(m) + i(n)$ となり，この関数は足し算を保存するので，準同型写像である．

事例 4.2

事例 2.5 の人生モノイド M の縮小版として，(1) がんばる, (2) がんばらない，という 2 つの選択肢と単位元 (i) のみから生成される自由モノイド M' を考える．M から M' への関数 f は，M の元に現れる「勉強する (b)」「バイトする (c)」を (1) に，「遊ぶ (a)」「休む (d)」を (2) に置き換えるものとすると，これは M から M' への準同型写像となる．例えば，$f(abd) \circ (dca) = 212 \circ' 212 = 212212 = f(abddca) = f(abd \circ dca)$ である．
(ところで自由モノイドは半順序を定義するのだった．M から M' へのモノイド準同型があるとき，そこから得られる半順序の間にはどのような関係があるか，考えてみても面白いかもしれない．)

練習問題 4.1

$(\mathbb{N}, +, 0)$ から $(\mathbb{R}^+, +, 0)$ への準同型写像には埋め込み以外にも沢山ある．その例を考えてみよう．

またこれまでと同様，モノイド M, N の間の準同型写像 $f : M \to N$ が全単射であるとき，f は**同型写像** (isomorphism) といわれる．M と N の間に同型写像があるとき，両者はモノイドとして**同型** (isomorphic) といわれ，これを $M \sim N$ と書く．ちなみにこのとき逆写像 $f^{-1} : N \to M$ も N から M への同型写像になっている．

事例 4.3

$\exp() : \mathbb{R}^+ \to \mathbb{R}^+$ の逆写像は，（e を底とする）対数関数 $\log()$ である．$\log(x) = y$ とは，e を y 乗すると x になる，ということを意味する．よって任意の $x \in \mathbb{R}^+$ につき，$\log(\exp(x)) = x$ であり，$\exp(\log(x)) = x$. また

$$\log(x \cdot y) = \log(x) + \log(y)$$

かつ

$$\log(1) = 0$$

より，log は掛け算 $(\mathbb{R}^+, \times, 1)$ から足し算 $(\mathbb{R}^+, +, 0)$ への準同型写像になっている．
一方で $(\mathbb{N}, +, 0)$ と $(\mathbb{R}^+, +, 0)$ の間には，当然全単射は存在しない．特に，事例 4.1 で見た埋め込み準同型は同型写像ではない．

モノイド同型 $M \sim N$ はモノイド間の同値関係である．まず恒等写像 $i :: m \mapsto m$ はモノイド同型なので $M \sim M$．また上に述べたように，$f : M \to N$ が同型写像（つまり $M \sim N$）なら $f^{-1} : N \to M$ が同型写像なので $N \sim M$．そして $f : M \to N, g : N \to O$ がそれぞれ同型写像なら，合成 $gf : M \to O$ も同型写像になる（気になる人は調べてみよう），つまり $M \sim N, N \sim O$ なら $M \sim O$.

事例 4.4

事例 2.3 で扱った文字列のモノイドと，事例 2.4 で見たロボットのプログラムは同型である．実際，前者をそのままロボットの動作プログラムと見ることができるし，また後者の動きから文字列を得ることができる．

練習問題 4.2

ロボットプログラムと人生モノイド（事例 2.5）の間に同型写像を作ることはできるだろうか．準同型写像はどうだろうか．

5 モノイド作用

本講ではこれまで，足し算や「選択」など，抽象的な操作や動作としてのモノイドの側面を強調してきた．この節では視点を変えて，そのような操作によって何らかの**対象がどう変化を受けるか**，という側面に目を向けてみたい．例えば事例 2.4 で見たロボットのプログラムでは，動作によってロボットの現在位置が変化していくはずだ．また人生モノイド（2.5）においては，選択によってあなた自身の状態が変わっていく．モノイドは，このようにある対象について作用を加えたときにその状態がどう変遷していくか，というダイナミックな過程をモデル化するためにも用いられる．その鍵になるのが，次に見る**モノイド作用** (monoid action) の概念である．

例としてロボットのプログラムを取り上げると，まずロボットの状態の集合 S を考えることができる．前進したり向きを変えたりといったモノイド元の一つ一つは，こうした状態を更新する S からそれ自身への関数である．これはつまり，命令と現在の状態の組から新しい状態へ

の関数を考えること，つまりモノイド演算を $M \times S \to S$ という 2 項関数として捉えるということにほかならない．

定義 5.1: モノイド作用

$(M, \circ, i)$ をモノイド，S を集合とする．モノイド M の集合 S への（左）M-**作用** *(M-act)* とは，写像

$$M \times S \to S, \quad (m, s) \mapsto ms$$

であり，以下を満たすものである：

1. 任意の $s \in S$ について，$is = s$.
2. 任意の $m, n \in M$ と任意の $s \in S$ に対して，$m(ns) = (mn)s$.

上の要件 1 は，単位元 i が状態を何も変化させない恒等写像 $i(s) = s$ であることをいっている．要件 2 は，モノイド元の結合が写像の合成になっており，ある状態 s に n を作用させてから m を作用させた結果と，モノイドの側で最初に合成した作用 mn を s に作用させた結果とが等しくなる，ということを述べている．

事例 5.1: ロボットへのモノイド作用

モノイド作用へのイメージを持つため，ロボットの事例をより具体化させてみよう．プログラムは一歩前進する (a)・右を向く (b)・左を向く (c)・何もしない (i) の 4 つの動作から生成されるのだった．これによって変化するロボットの状態は，その平面上の位置と東西南北の向きである．前者は整数の組 $(x, y) \in \mathbb{Z} \times \mathbb{Z}$，後者は東西南北で表すことができるだろう．これらを合わせてロボットの状態集合は $S = \mathbb{Z} \times \mathbb{Z} \times \{w, e, s, n\}$ と定義できる．
いま現在のロボットの状態を $s = (x, y, d)$ とし，これに前進する (a) というモノイド元を作用させると，次の状態は as となる．x 要素についてこれを具体的に書くと，次のようになるだろう．

$$a(x) = \begin{cases} x + 1 & \text{if } d = w, \\ x - 1 & \text{if } d = e, \\ x & \text{otherwise.} \end{cases}$$

つまりロボットが東を向いているときは東に，西を向いているときは西にそれぞれ一歩進み，南面・北面しているときは x 座標は変わらない．

練習問題 5.1

同様に，$a(y), a(d), b(x), b(y), b(d), c(x), c(y), c(d)$ がどのように計算されるかを示し，モノイド作用の全貌を明らかにせよ．

このロボットの例のように，作用モノイドが有限の元（ここでは a, b, c, i）から生成される場合，生成元の作用のみを定めれば，モノイド全体の作用が定まる．というのも，モノイド作用の要件 2 より，合成された作用の結果は，各作用を繰り返し適用することで得られるからだ．例えば $aabc(s) = a(a(b(c(s))))$ のように求めることができる．

以上では命令によって状態が変わる様をモノイド作用としてモデル化したが，より能動的に，

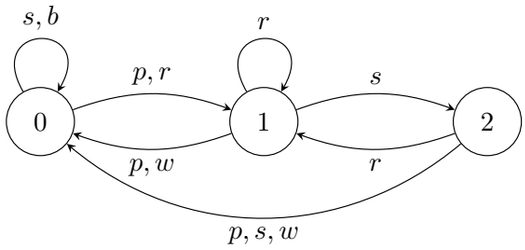

図 2　精神状態の遷移図．それぞれのノードから p, s, w, r の 4 つの経路が出ていることを確認せよ．

エージェントが動作や外からの刺激によって状態を変えていく様を表現することもできる．

> **事例 5.2: 状態遷移**
>
> 2.5 で日々の選択を考えたが，こうした選択は当然動作主であるあなたの状態を変化させるだろう．その変化をモノイド作用としてモデル化してみよう．主体の精神状態として，(0) 疲労，(1) やる気，(2) 充実，の 3 つを考える．(p) 遊ぶ，(s) 勉強する，(w) バイトに行く，(r) 休むという 4 つの作用によってこれらの状態がどう変わるかは，以下のような表で表すことができる．
>
> | | p | s | w | r |
> |---|---|---|---|---|
> | 0 | 1 | 0 | 0 | 1 |
> | 1 | 0 | 2 | 0 | 1 |
> | 2 | 1 | 0 | 0 | 1 |
>
> 表の各エントリーは，一行目の各作用が，一番左端に示される状態に作用したときに変化する先の状態を示している．例として一番左の列を考えると，これは遊ぶ (p) という作用は疲れているときはやる気に，やる気のときは疲れに，充実しているときはやる気にへと状態を変化させる，ということを表している．なお単位元 i はいかなる状態についても $is = s$ と変えないので省略してある．

ちなみに同様の演算は，図 2 のような**状態遷移図** (state transition diagram) によっても表すことができる．状態集合を持ち，その間の状態遷移関係が定まっているような機械を，**オートマトン** (automaton) という．オートマトンは，状態集合へのモノイド作用として考えることができる．

> **事例 5.3: モナド**
>
> 哲学において，このような考えは物の本性を考える上で大きな役割を果たしてきた．例えば『モナドロジー』においてライプニッツは，モナドを「非物体的なオートマトン (des Automates incorporels)」と呼び，アリストテレス的なエンテレケイアと同一視している．というのもライプニッツにとって，モナドとは固有の行動原理・プログラムを内部に持ち，それを展開していく存在にほかならないからだ．つまりモナドとは，一個のモノイドとして規定できると言える．

このように考えると，モノイド作用の射程は極めて広く，時間発展する系一般を記述することができる．それには例えばコンピュータのプログラムの動作や，物理系の時間発展などが考えられる．例えば古典力学の法則は，各質点の位置と運動量からなる**状態空間** (state space)

に対するモノイド作用として考えることができる．こうした時間発展系を**力学系** (dynamic systems) という．

この定義の集合 X を M にすると，モノイドの定義 2.1 が復元されることを確認しよう．つまりすべてのモノイドは自分自身へのモノイド作用だといえる．[*1]

これまで見てきた「操作」としてのモノイドは，原則的にすべてこの「対象への作用」という面から捉え直すこともできる．例えば足し算のモノイドでは，演算 $+$ は 2 つの整数 (e.g., $5, 7$) をとって 1 つの整数 (12) を返す 2 項関数なのだった．しかし同様の事態を，全てのモノイド元 m は，他のモノイド元 n に「m を足す」1 項関数，すなわち $m : M \to M, n \mapsto m + n$ である，と表すこともできるだろう．これはつまり，例えば具体的な数 7 を，「7 を足す」作用として考えよう，ということだ．モノイド元の合成は作用の合成，例えば $m + n$ は「m を足してから n を足す」という一つの作用となり，また単位元 0 は「0 を足す」(つまり何も変えない) という作用である．このようにモノイドは，「自分自身に働きそれを変える作用」として考えることができる．

> **事例 5.4**
>
> 脳に加えられる外的・内的刺激全体を M とすると，M は脳状態へ作用するモノイドと捉えられる．ここで $m, n \in M$ に対しその合成 mn は，「刺激 m を加えたあとに刺激 n を加える」こととする（ただし「何も刺激を与えないこと」を単位元と考えるのは良くないかもしれない．その問題点を考えてみよ)．

> **練習問題 5.2**
>
> モノイド作用もしくはオートマトンの例を挙げよ（自分で創作しても良い)．その際，モノイド元と状態集合，および作用の積表を明示すること．

参考文献

[*1] 左作用についての注.