解　説

ニューロ・コンピューティング
—原理と概要—†

麻　生　英　樹††

1. はじめに

現在の計算機は，脳の行っていることを参考に作られているが，脳の物理的な構造，働き方はほとんど参考にしていない．これに対して，脳のしていることを機械化するのであれば，脳の構造や働き方を参考にするのがよいのではないかという考えも自然である．ニューロ・コンピューティングの研究は，こうした考え方に基づいて，脳の構造と類似したメカニズムによる情報処理の可能性を探り，現在の計算機とは違った特性をもつ情報処理装置の基礎となる情報処理様式を確立するとともに，その情報処理様式を具体的な個々の情報処理の問題に応用するための応用技術を獲得してゆく試みであるといえるだろう．

この場合，脳の構造，働き方をどの程度取り入れるかという点に関してさまざまな方針がありうる．また，脳，あるいは神経系全体を含む生体の情報処理のメカニズム自体もいまだに解明されてはおらず，研究の対象である．このような理由によって，ニューロ・コンピューティングの研究は，非常に単純な脳のモデル化に基づくものから，神経細胞の詳細な性質の解明をめざすものまで，かなり幅広いものになっている．また，メカニズムの構造が決まったとしても，そうしたメカニズムを使ってどのような情報処理をどのように行わせるのかも問題である．これについても非常に単純な情報処理から人間の行っているような高度な情報処理まで幅広い研究が行われている．

こうした，ニューロ・コンピューティングの研究によって得られた一つの情報処理様式を工学的にハードウェアによって実現したものがニューロ・コンピュータということになる[1]．本稿では，ニューロ・コンピュータの物理的実現に関する解説への序として，そのための基本となる情報処理様式，すなわち，ニューロ・コンピューティングの研究の現状を中心に概説する．

2. ニューロ・コンピューティング

すでに述べたように，一口にニューロ・コンピューティングの研究といってもその内容は多岐にわたっている．最大公約数的な枠組みとしては，多数の計算要素（ニューロン）の相互作用によって情報処理を行う，というくらいのことしかいえない（少なくとも，さまざまな研究が方向を模索しながら行われている現在では，はっきりとした枠組みを想定するのは得策ではない）．たとえば，ニューロンのモデルにしても，単純なしきい素子のようなものや，複雑な非線形の偏微分方程式で記述されるものなどさまざまである[2),3)]．こうした研究のすべてを紹介することはできないため，ここでは，いくつかの視点から代表的と思われる研究を紹介することによっておおよその全体像を描くという分散的な概説を試みる．また，以下では具体的なモデルを紹介するのみであるが，そうしたモデルのさらに基礎には，生理学，解剖学，薬理学，心理学，病理学などの多くの研究があることも忘れてはならない[1)]．

2.1 ニューロンのモデル

現在のほとんどすべての研究において，ニューロンは多入力1出力の素子であると考えられている．出力値のほかに内部状態をもつ場合もある．以下，ニューロンのモデルとして代表的なものをあげる．

（1） しきい素子（McCulloch and Pitts のニューロン）[4)]

ニューロンのモデルとして最も古典的で単純なもの．初期のパーセプトロン（Perceptron）でもこのモデルが使われている[5)]．内部状態なしの二値のしきい素子であり，自分への入力の総和を計算して自分のしきい値と比較して，入力の総和がしきい値よりも大き

† Neurocomputing: An Overview by Hideki ASOH (Electrotechnical Laboratory).
†† 電子技術総合研究所

い場合には値1を出力，そうでない場合には値0を出力する．

（2） 準線形素子

ニューロンのモデルのなかでも最もよく使われているもの．内部状態なしで連続値の出力をとる．自分への入力の総和（しきい値を含む）を計算して，その値を自分の特性関数（この関数は普通，入出力関数などと呼ばれるが，ここでは，多入力-1 出力の素子であるニューロン全体の入出力関係を表す関数と区別するためにこう呼ぶことにする．）によって変換して出力値を得る．特性関数としては，通常，ロジスティック関数

$$f(x)=\frac{1}{1+\exp(-x)} \qquad (1)$$

などのシグモイド状の関数が使われる．これは，実際のニューロンの飽和的な入出力特性を反映させたものである．特性関数としてステップ関数を使ったものが(1)に対応する．

（3） 微分方程式/差分方程式モデル

ニューロンの状態変化を時間発展する微分/差分方程式によって記述したもの．使われる方程式はいろいろあるが，代表的なものは，

$$\begin{cases}\dfrac{du}{dt}=-\dfrac{1}{\tau}u+i\\ v=g(u)\end{cases} \qquad (2)$$

ただし，u はユニットの内部状態値，v は出力値，i はユニットへの入力の総和，g はシグモイド状などの連続関数

というような，入力の総和の値による出力値の増大と時間による忘却効果，飽和効果などを盛り込んだものである[6]．広くいえば，(1)，(2)のモデルも差分方程式モデルに含まれる．最近では，カオスを発生することのできる方程式を使っている場合もある[1]．

また，ニューロンの軸索部，樹状突起部，などそれぞれの部分に関しての詳細な微分方程式モデルなども存在するが，こうした詳細なモデルは，現在のところ，ネットワークを作る場合にはほとんど用いられていない．

（4） 確率的なモデル

ニューロンの動作を確率的な状態変化規則，確率微分方程式などでモデル化することもある．たとえば，ボルツマン・マシン (Boltzmann Machine) と呼ばれるニューラル・ネットワークのモデルでは，ユニットへの入力の総和 i に対して，

$$p=\frac{1}{1+\exp(-i)} \qquad (3)$$

という式から定まる確率で状態値（出力値）を1にするという確率的な状態変化規則が採用されている[7]．

2.2 ニューラル・ネットワークのモデル

個々のニューロンを組み合わせて一つのネットワークを作るためには，ニューロン間の相互作用をモデル化する必要がある．実際の生体のニューロン同士の相互作用にはいまだ不明な部分が多いが，最も主要な相互作用と考えられているのは，シナプスという接合をとおして一つの細胞がもう一つの細胞へ興奮を伝達するという作用である（厳密にいうと，シナプスにもいろいろな種類があり，必ずしも一方向の伝達ばかりではないらしい）．

現在のほとんどのニューラル・ネットワークのモデルにおいても，このシナプスを通じての作用がモデル化されている．なかでも最も多いのは，それぞれのシナプスに信号の伝達効率にあたる実数 w を割り当て，そのシナプスを通過する信号は w 倍されて相手のニューロンに伝達される，というもので，この実数のことを「シナプスの荷重」，あるいは，簡単に「重み」などと呼ぶ．w は負の値をとることもある．例として，ニューロンに準線形の素子を使った場合を示す(図-1)．

このような準線形のニューロンを重み付きの結合でつないだネットワークが，現在までのニューロ・コンピューティングの主流である．このモデルにおいては，それぞれのニューロン（ネットワークのなかのニューロンのことをノード，ユニットなどとも呼ぶ）で行われる計算は，ベクトルの積和（内積）演算とニューロンの特性関数の計算ということになる．こうした単純なモデルのほかにもシナプスの物理的形状と伝達効率の関係の詳細なモデルなどいろいろなモデルがある．

シナプス結合による相互作用というモデルを採用したとして，ニューロンをどのように結合してネット

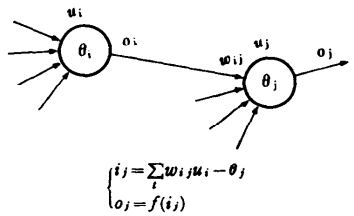


図-1 準線形素子のネットワーク

ワークを作るかに関しては，特別な制約があるわけではない．しかし，いくつかの典型的な結合のしかたが存在し，それぞれの特徴をもっている．

（1） 階層型のネットワーク

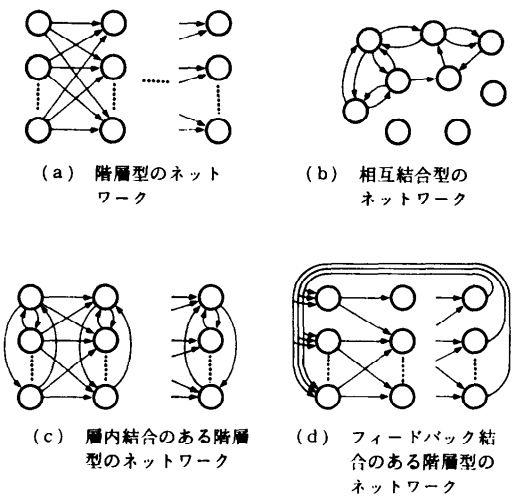
ユニットがいくつかの層に分かれており，それぞれの層に順番がついていて，各層は自分よりも前の層からしか入力を受けない．したがって，前に戻るフィードバックの結合がないため，第1層（入力層と呼ばれる）に与えられた入力パターンが次々と層を経ながら変換されて最終層（出力層と呼ばれる）に達することで1回の動作が終了する．

（2） 相互結合型のネットワーク

任意の二つのユニットの間に互いに結合があるようなネットワーク（完全結合なネットワーク）．フィードバックの結合があるため，ネットワークの動作はダイナミクスをもつ．ある初期状態から出発したネットワークは状態変化を繰り返すうちに平衡状態（attractor）に達するが，ネットワークの構造，ユニットの性質などによって，静的な平衡状態，振動，カオス的な平衡状態など，さまざまな平衡状態が現れる．こうしたダイナミクスの解析には力学系の理論などが使われるが，一般に非線形の力学系になるため，解析はそれほどやさしくない．

この二つのタイプが軸の両極であり，一部のユニットの間にだけ相互結合がある場合，階層型で，最終層から入力層へのフィードバック結合があるものなど，さまざまな中間形が存在する（図-2）．

もう一つ，ネットワークを組む場合に考慮されるのは，それぞれのニューロンが動作するタイミングの問題である．これについては，（1）すべてのニューロンが同一のタイミングで状態変化を行う．（2）それぞれのニューロンが独立に自分のタイミングに従って状態変化を行う．（3）信号の伝達にかかる時間などもモデル化して，信号の伝達とともに，ニューロンが状態変化を行っていく，などの方法が考えられ，モデルの特性などによって適当な方法が選ばれている．



（a） 階層型のネットワーク　（b） 相互結合型のネットワーク

（c） 層内結合のある階層型のネットワーク　（d） フィードバック結合のある階層型のネットワーク

図-2 いろいろなネットワーク

ネットワークに組まれたニューロンが相互作用しながら状態変化を繰り返していくさまは，ある種のオートマトンを思わせる．実際，ニューラル・ネットワークの研究の初期に，McCulloch-Pitts 型のニューロンを使ったネットワークに関して，ネットワークと有限オートマトンが等価であるという結果が知られている[8]．

2.3 ニューロ・コンピューティングのモデル

こうして，ネットワークのモデルができあがると，次は，それを使った情報処理＝「ニューロ・コンピューティング」のモデルである．これについてもさまざまな提案がなされている最中であるが，現在のところ最も代表的な使いかたとしては次の二つをあげておけばよいだろう．

（1） パターン変換，時系列変換

ネットワークの一部を入力部，出力部として，入力部に与えたパターンを変換して出力部に結果のパターンを出す，というような使い方．あるいは，入力部にパターンあるいは時系列を入力して，出力部に時系列を出力させることもある．おもに，パターン認識，連想記憶，運動系列の生成，などの実現に使われている．

典型的な例としては，階層型のネットワークによるパターンの変換がある．前にも書いたように，階層型のネットワークにおいては，入力層に与えられたパターンは次々と層を伝播されてゆきながら変換され，最終の出力層に達する．これで1回のパターン変換が終了する．ここで，ある入力パターンに対する出力がそのパターンの属するクラスの表現である場合には，ネットワークはパターン認識を行っていることになるし，入力パターンが連想のための鍵入力，出力が鍵から想起されるパターンである場合には，連想記憶を実現しているということになる．

ニューロンのタイプを指定すると，一つの層を通過する間に行える変換処理が限定される．したがって，ある変換を実現するのに何層くらいのネットワークが必要であるか，というような問題が生じる．たとえ

ば，n変数の論理関数の計算をn個の入力ユニットと1個の出力ユニットをもつ階層的なネットワークに行わせることができるが，これについて，しきい素子型のニューロンを使った場合には，3層のネットワークによって任意の論理関数を計算できることが知られている．例として，排他的ORを計算するネットワークを示す（図-3）．

連続な出力値をとるユニットのネットワークによってn入力の連続関数を近似させることもできるが，これについて，準線形で，シグモイド型の特性関数をもつニューロンを使った場合には，3層のネットワークがあれば，任意の連続関数を任意の精度で近似できることが知られている[9]．ただし，一般に，精度をあげるためにはかなりたくさんの中間の層のユニットが必要である．

また，Minsky と Papert は，各層の間の結合の数が制限されている階層的ネットワークの論理関数計算能力について考察している．そこでは，それぞれのニューロンへの入力結合の数を制限した場合には計算できないような論理関数の例などが示されている[10]．

パターンの変換を行わせるもう少し別な方法として，相互結合型のネットワークを利用する方法もある．この場合には，ネットワークの一部を入力パターンによって固定したままネットワークを動作させて，ネットワークが平衡状態に達したときの出力部の値を出力パターンとする．階層的なネットワークによるパターン変換が，比較的連続な，すなわち，似ている入力パターンには似た出力パターンが対応するような変換課題を得意とするのに対して，こちらは，比較的不連続な，入力パターンの少しの変化が出力パターンの大きな変化を生むような変換課題に向いていることが予想される．

（2） 最適化機械

これは Hopfield らによって有名になった使い方である[11]．ある種の相互結合型のネットワークでは，ネットワークが状態変化するとともに必ず減少／増加するようなネットワークの状態の関数を作ることができる．これを，統計力学とのアナロジから，ネットワークのエネルギ関数と呼ぶ（力学系の理論ではリアプノフ関数と呼ばれている）．したがって，解きたい最適化問題の変数をユニットの状態値によって表現し，評価関数がこのエネルギ関数に一致するようにネットワークの結合を決めることができれば，適当な初期状態からこうしたネットワークを動作させることによって，評価関数を最小／最大にするような変数の値の組み合わせを求めることができる．こうした方法は，巡回セールスマン問題，グラフの対応付けなどの組み合わせ最適化問題やパターンの一部から全体を想起するような連想記憶などに応用されている．また，ある固定された入力パターンに対して評価関数を最小／最大にするような出力パターンを求める，というように，パターンの変換を最適化問題として定式化することもできる．

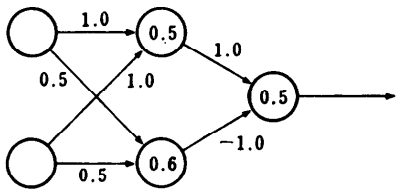


図-3 排他的 OR を計算するネットワーク
○の中の数字はユニットのしきい値，
→の付近の数字は結合の重みを示す．

具体例として，Hopfield のネットワークを用いて巡回セールスマン問題を解く場合を紹介する．Hopfield のネットワーク（離散値のもの）は対称な相互結合をもった二値のしきい素子から成るネットワークである．ユニットiとjの間の結合の重みを$w_{ij}(=w_{ji})$，ユニットiのしきい値をθ_iとすると，ユニットiの状態変化規則は，

$$\begin{aligned}&\sum_j w_{ij}u_j-\theta_i>0 \text{ ならば状態を1にする}\\&\sum_j w_{ij}u_j-\theta_i<0 \text{ ならば状態を0にする}\\&\sum_j w_{ij}u_j-\theta_i=0 \text{ ならば状態を変化させない}\end{aligned} \quad (4)$$

と書ける．このとき，

$$E=-\frac{1}{2}\sum_{i\neq j}\sum w_{ij}u_iu_j+\sum_i \theta_i u_i \quad (5)$$

という，ユニットの状態値の関数を考えると，この関数は上の規則に従う状態変化の際に，（状態が変化しない場合を除いて）必ず減少するということが示せる．したがって，このネットワークを適当な初期状態から動作させると，ネットワークは必ずある静的な平衡状態に達し，その平衡状態はEの極小値を与えるものになっている．

こうしたネットワークで巡回セールスマン問題を解くには，問題の変数をユニットの状態値によって表現する必要がある．巡回セールスマン問題にはいろいろなヴァリエーションがあるが，ここでは，N都市を一度ずつ巡ってもとの都市に戻るような最短の経路を求める問題を考える．この場合，問題の変数はそれぞれ

の都市を一度ずつ巡る経路である．Hopfield らは次のような方法で一つの経路を表現した．

$N\times N$ の矩形に並んだ N^2 個のユニットをもつネットワークを用意する．ユニットの並びの行を各都市に，列を順番に対応させる．こうすると，一つの経路を，各行各列に一つずつ値1をとるユニットがあるようなネットワークの状態によって表現することができる(図-4)．このとき，その経路の長さをユニットの状態値の二次式で書くことができる．したがって，その二次式が式(5)のエネルギ関数に一致するように結合の重みを決めることができる．実際には，ネットワークの状態を経路を表現するものに限るための罰金項を加えた次のようなエネルギ関数を使う．

$$E=\frac{1}{2}\sum_{i}\sum_{j\neq i}\sum_{k}d_{ij}u_{ik}(u_{jk+1}+u_{jk-1}) \qquad (6)$$
$$+A\left(\sum_{i}\left(\sum_{k}u_{ik}-1\right)^2+\sum_{k}\left(\sum_{i}u_{ik}-1\right)^2\right)$$

ただし，u_{ik} は i 行 k 列にあるユニットの値，d_{ij} は都市 i と j の距離である．また，A は距離と罰金のバランスを決めるパラメータ．このとき，u_{ik} と u_{jl} の間の結合の重み $w_{ik\,jl}$ と θ_{ik} は，

$$w_{ik\,jl}=-d_{ij}(\delta_{lk+1}+\delta_{lk-1}) \qquad (7)$$
$$-A(\delta_{ij}(1-\delta_{kl})+\delta_{kl}(1-\delta_{ij}))$$
$$\theta_{ik}=-A/2$$

ただし，δ_{ij} は

$$\delta_{ij}=\begin{cases}1 & i=j\\ 0 & i\neq j\end{cases}$$

となる．

一般に関数 E は複数の極小値をもつため，この方法によって必ず E の最小値を与える経路が求められたわけではない．E の値がどの極小値に収束するかは，どのような初期状態から出発するかに依存する．Hopfield らは，ランダムに初期状態を選んで何度かネットワークを動作させ，得られた解のなかで最小のものを使うことを提案しているが，最近，上坂は，連続値のユニットを使ったネットワークに関して，E がある条件を満たす場合には必ず最小値を与えるような初期値の選び方を示唆している[12]．

2.4 プログラミングと学習のモデル

ネットワークが実際にどのような働きをするかを決めているのは，個々のニューロンの特性とニューロンの結合のしかたである．最も普通なモデルを使っている場合には，これは，ニューロンの特性関数の形と，それぞれの結合に付与されている結合の重みの値，ということになる．したがって，ネットワークに特定の

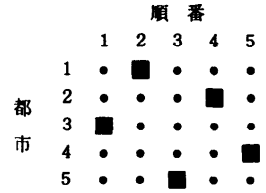


図-4 巡回セールスマン問題を解くためのネットワークと経路の表現
■は値1のユニット．
(3→1→5→2→4→3) という経路が表現されている．

情報処理をさせるためには，こうしたネットワークの動作を決めるパラメータを適切に設定しなくてはならない．このパラメータ設定の作業が従来の計算機でのプログラミングにあたる，ということもできる．

こうした設定はそれほどやさしいものではない．現在のところ，大きく分けて2通りの方法が使われている．一つ目は，ネットワークに解かせる問題の定義から結合の重みが計算できる場合である．たとえば，評価関数を最適化する場合には，評価関数がネットワークのエネルギになるようにするための結合の重みがあらかじめ計算できる．上で示した巡回セールスマン問題を解く場合にもこの方法を使っている．

もう一つは，あらかじめ適切な結合の値が分からない場合，あるいは，分かっていても一つ一つ設定するのがめんどうな場合に使われる方法で，「学習」と呼ばれている方法である．おおざっぱにいえば，ニューラル・ネットワークの学習とは，ネットワークにいろいろな例題を解かせて，その結果を使いながら，処理がうまくゆくような方向にそれぞれの結合の重みを少しずつ修正してゆくことによって，最終的に適切な結合の重みを得ようというものである．こうしたシナプス結合部の伝達効率を変化させて，ネットワークの振舞いを変化させるという技法は，生体内でも使われている．

こうした学習による結合の設定の典型的な例としては，階層的なネットワークによってパターン変換をさせる場合がある．この場合には，学習のための訓練例として入力パターンとそれに対する正解出力パターンの組みをたくさん用意しておき，ネットワークに，そのなかから選んだ適当な入力を与えて，出力を計算させ，その結果を正解の出力と比較する．誤差がある場合には，その誤差の情報を利用してネットワークの結合を修正する．これをたくさんの例について繰り返すことによって，最終的に，常に正解を出すような結合を得るわけである．誤差の情報からそれぞれの結合の

修正量を導くためのアルゴリズム（「学習のアルゴリズム」と呼ばれる）としては，さまざまなものが提案されているが，そのなかでも最も有名なのが，パーセプトロンの学習法と，その拡張である誤差逆伝播アルゴリズムである[13]．以下，学習のアルゴリズムの例として，誤差逆伝播学習のアルゴリズムを簡単に紹介する．

M層の階層的なニューラル・ネットワークを考える．第k層の第iユニットをu^k_iとする．それぞれのユニットは準線形であるとする．をi^k_i u^k_iへの入力の総和とし，fをユニットの特性関数，o^k_iをu^k_iの出力値とする．$w^k_i{}^{k+1}_j$を第k層の第iユニットから次の$k+1$層の第jユニットへの結合の重みとし，θ^k_iをk層の第iユニットのしきい値とする．ただし，以下の議論では，各層に一つ（ネットワーク全体に一つでもよいが，記号の都合でこうする），常に値-1を取る特殊なユニットu^k_0を置き，各ユニットのしきい値θ^{k+1}_jをそのユニットからの結合の重み$w^k_0{}^{k+1}_j$に帰着させることにする．こうすることで，しきい値に関する学習規則を結合の重みに対する学習規則に帰着させることができる．

学習に使う訓練用の入出力パターンの組みとして，入力パターンとその入力に対する望ましい出力パターンの組みの集合$\{(\boldsymbol{x}_p, \boldsymbol{y}_p)\}$$(p=1,\cdots,N)$が用意されているとする．いま，適当な入力パターン$\boldsymbol{x}$を入力したときの出力層の第$j$ユニットの出力を$o^M_j$とし，対応する正解出力パターン$\boldsymbol{y}$の第$j$成分を$y_j$とする．

このとき，誤差逆伝播学習では，次の式に従って各結合の重みを修正する．

$$w^{k-1}_i{}^k_j = w^{k-1}_i{}^k_j + \eta\delta^k_j o^{k-1}_i \qquad (8)$$

$$\delta^M_j = (y_j - o^M_j) f'(i^M_j) \qquad (9)$$

$$\delta^{k-1}_i = f'(i^{k-1}_i)\sum_j w^{k-1}_i{}^k_j \delta^k_j \quad (k=M,\cdots,3)$$

式(8)が修正式であり，ηは小さな正の数である．δ^k_jを一般化された誤差などと呼ぶ．式(9)の二つ目の式は，δ^k_jを，ちょうど，出力層での誤差δ^M_jを出力層への入力として，出力層から入力層の方向へ逆方向に（$f'(i^k_j)$を掛けながら）伝播しているような形で$k=M$から3まで順次計算するための漸化式である．これが誤差逆伝播という名前の由来である．

ユニットの特性関数としてロジスティック関数を用いている場合には，$f'(i^k_j)$は$o^k_j = f(i^k_j)$の値を使って

$$f'(i^k_j) = o^k_j(1-o^k_j) \qquad (10)$$

として計算できる．

修正式(8)は，ユニットu^k_jへの結合を，u^k_jの一般化された誤差δ^k_jとその結合が伝えている信号o^{k-1}_iとの積に応じて修正するものであり，誤差が正，すなわち，u^k_jの活動が足りない場合には，u^k_jへ正の信号を送っていたユニットからの結合を増やし，負の信号を送っていたユニットからの結合を減らすように働くことが分かる．誤差が負の場合には結合の修正も逆符号になる．

この学習法によって最終的に得られる結合は，訓練セットに関して，ネットワークの出力と正解との2乗誤差の総和

$$R = \sum_p (\boldsymbol{y}_p - \boldsymbol{o}^M(\boldsymbol{x}_p))^2$$

を極小にするものであることが知られている．また，連想記憶のための学習法として知られている直交学習は，誤差逆伝播学習を2層のネットワークに使った場合である．

このほかにも，相互結合型のネットワークであるボルツマン・マシンの学習のアルゴリズム[14),15)]，それぞれのユニットの競合によってユニットの働きの分業を実現する競合型の学習[16),17)]などいろいろな学習法が提案され，それぞれの目的に応じて使われている．

こうした一般的な学習能力は，「学習させればなんでもできるようになる」というように過剰に評価されがちであるが，原理的な能力と実際的な能力とのギャップは（現在の計算機のそれと同様あるいはそれ以上に）きわめて大きい．そもそもこうした単純なアルゴリズムによる学習が可能であるのは，ニューラル・ネットワークの要素処理の簡単さと構造の単純さによるところが大きいわけであり，複雑な問題の解決法を学習することの困難さと裏腹な関係にあるといえる．したがって，こうした学習能力を含めたネットワークの情報処理能力をうまく活かせるような問題の選択と分割，設定が重要である．

2.5 ニューロ・コンピューティングの応用

2.3で述べた，パターン変換と最適化をとりあえずのニューロ・コンピューティングの基本機能とすれば，実際の情報処理の問題を解決するには，それらを組み合わせて使うための応用技術が必要になるであろう．こうした応用技術は，さまざまな具体的問題を解決してゆくなかで生まれてくるべきものである．また，そうした研究のなかで，新たな基本機能や，より強力なニューラル・ネットワークのモデルが作られてゆくことが望まれる．

現在のところでは，ニューロ・コンピューティング

の応用対象は，その基本機能をそのまま使うことによって実現できるような種類の問題—具体的には，音声，画像などのパターン情報の認識と処理，連想記憶，多変数制御，数量的推論，制約のあるラベリング，マッチングなど—が主流である[18]．もちろん，こうした基本機能によって実現できる問題はニューロ・コンピューティングにとって最も得意な問題であるはずであり，まずそうした領域において，その並列性，学習能力といった特徴を活かして，現在の計算機を凌ぐような具体的な成果をあげることはとても重要である．一方，実用的な応用には遠いが，コネクショニズム，PDP などの名前のもと，認知科学などの文脈において，ニューラル・ネットワークを用いて，論理的推論，逐次的な手続き処理，自然言語操作，認知地図操作などの情報処理を実現しようという試みも盛んに行われている[17]．そこでは，従来の計算機の上ではラベルとしてしか実在しなかった記号をもう少しミクロなレベルから組み立てることによって，記号にもう少し豊かな実体を与えようというような試みがなされたりしている．

こうした応用研究において重要なことは，従来から行われているそれぞれの分野における情報処理の研究の成果をうまく取り入れることであろう．そうした成果のなかには，現在の計算機の構造に深く依存するものもあるが，その一方で，ある情報処理を行うために，どのような情報が不可欠であるか，また，その情報処理で作られる情報はどのような性質をもつべきであるかなど，計算機の構造に依存しない結果も多い (D. Marr の計算理論[19])．ニューロ・コンピューティングのもつ学習能力は，しばしばこうした従来の結果を無視し，すべてを学習でまかなおうという試みを産みやすいが，学習能力に頼りすぎることは避けるべきであろう．必要なのは，馬鹿な万能ネットワークではなく，それぞれの情報処理の問題の本質的な性質を上手に体得している，賢い，かなり分化の進んだネットワークなのである．もちろん，こうした分化したネットワークを作る場合に，学習能力がヒントを与えることもあるとは思われるが，それよりも，問題そのものを分析することによって得られる知識が，ネットワークを構築する場合にも役立つはずである．

2.6 シミュレーションとハードウェア化

パターン認識などの情報処理において，その能力を現在の方法との比較のまな板にのせるためにも，また，推論，自然言語操作などの情報処理の実験においてかなり大規模なネットワークを気楽に使えるようになるためにも，ニューロ・コンピューティングの特徴を活かせるハードウェア＝ニューロ・コンピュータの開発が強く望まれるところである．現在ほとんどの応用研究では従来の計算機上のソフトウェアによるシミュレータが使われている．そうしたシミュレータの性能はいろいろであるが，Sun 3 などの普通のワークステーション上で動くもので，速度が 0.1〜数 MCPS (Mega Connection Per Second)，ネットワーク規模の上限が数万〜数百万結合くらいである（たとえば，筆者は StarNet というパブリック・ドメインのソフトウェアシミュレータ[20]を Aliant 社の FX-8 という計算機上で小規模並列に動かして使っているが，各層 100 ユニット (特性関数はロジスティック) で 3 層の階層的ネットワーク (20,000 結合) に 1,000 回パターン変換させるのにかかる CPU 時間は約 18 秒であった．ただし，このプログラムは FX-8 向けに特別にチューンされているわけではない)．パーソナル・コンピュータやワークステーションに内積計算などを高速にするための専用のハードウェアを付加したような製品もいくつか存在する[21]．さらに，ベクトル計算機，並列計算機，コネクション・マシンのような Massively Parallel な計算機，など既存のさまざまな超高速計算機を使ったシミュレータの実験開発も行われているが，やはり，ニューロ・コンピューティングに向いたハードウェアが開発されることが望ましい．現在応用研究に使われているネットワークの規模は，階層的なネットワークの場合で，1 層が数十から数百ユニット程度，3〜5 層というようなものが多い．この程度の規模のネットワークでも，しっかりと学習できるものが比較的安価なハードウェアで実現されて手軽に使えるようになれば，かなり大きな影響があるだろう．

ハードウェア化の試みについては本特集の他稿において詳しく解説されるが，大きく分けると，直接的な方法と間接的な方法の二つがある．直接的な方法とは，文字どおり，ネットワークを直接ハードウェア化するものであり，一つ一つのニューロンに対応するメカニズムを物理的に実現し，結合も実際に一つ一つ作るというものである．一方，間接的な実現は，適切な計算メカニズムの上で，仮想的なネットワークを実現する方法である．もちろん，結合だけを仮想にするなどの中間的な方法もある．デバイスとしては，従来技術の延長としてのシリコン素子，光-電子素子，光素子，そして分子素子などが検討され始めている．

3. おわりに

ニューロ・コンピューティングの研究およびニューロ・コンピュータの実現の研究の歴史は現在の計算機と同じくらい，あるいはそれ以上に古いともいえる．しかし，やっとここにきて本格的な研究開発が始まろうとしているところである．その最大の背景が，現在の計算機技術の成熟によってネットワークのシミュレーションがかなり手軽にできるようになったことだ，というのはおそらく事実であろう．電子計算機の第1号が作られてから約 40 年，計算機は爆発的に進歩普及し，その用途も大幅に拡大した．後発であるニューロ・コンピュータの前途はこれよりもかなり険しいものかもしれない．しかし，ハードウェア（物理的実現）的にもソフトウェア（情報処理様式とその使用）的にもさまざまな新しい技術を必要とするであろうこの道は，十分に魅力ある道であり，また探求する価値のある道であると思われる．

参考文献

(詳しい文献表は，1)〜3)，13)，18) などを参照)

1) 合原一幸：ニューラルコンピューター脳と神経に学ぶ—，東京電機大学出版局 (1988)．
2) 甘利俊一：神経回路網の数理—脳の情報処理様式—，産業図書 (1978)．
3) 福島邦彦：神経回路と自己組織化，共立出版 (1979)．
4) McCulloch, W. S. and Pitts, W.: A Logical Calculus of the Ideas Immanent in Nervous Activity, Bullet. Math. Biophysics, Vol. 5, pp. 115-133 (1943).
5) Rosenblatt, F.: The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, Psychol. Rev., Vol. 65, No. 6, pp. 386-408 (1958).
6) Hopfield, J. J.: Neurons with Graded Responce Have Collective Computational Properties Like Those of Two-State Neurons, Proc. Natl. Acad. Sci. USA, Vol. 81, pp. 3088-3092 (1984).
7) Hinton, G. E., Sejnowski, T. J. and Ackley, D. H.: Boltzmann Machines: Constraint Satisfaction Networks That Learn, Tech. Rep., CMUCS-84-119, Carnegie-Mellon Univ. (1984).
8) Kleene, S. C.: Representation of Events in Nerve Nets and Finite Automata, Automata Studies (Annals of Mathematical Studies, No. 34), Princeton (1956).
9) 舟橋賢一：ニューラル・ネットワークの capability について電子情報通信学会技術研究報告．MBE 88-52 (1988)．
10) Minsky, M. and Papert, S.: Perceptrons, Cambridge, MIT Press, MA: (1969) (斉藤正男訳：パーセプトロン，東京大学出版会 (1971))．
11) Hopfield, J. J. and Tank, D. W.: Neural Computation of Decisions in Optimaization Problems, Bilo. Cybern., Vol. 52, pp. 141-152 (1985).
12) 上坂吉則：二値変数の実数値関数から導かれるエネルギーを持つニューロン回路網の安定性について，電子情報通信学会技術研究報告，PRU 88-6 (1988)．
13) Rumelhart, D. E., Hinton, G. E. and Williams, R. J.: Learning Internal Representaions by Error Propagation, In Parallel Distributed Processing, Vol. 1, McClelland, J. L., Rumelhart, D. E. and The PDP Research Group, Cambridge, MIT Press, MA (1986).
14) Ackley, D. H., Hinton, G. E. and Sejnowski, T. J.: A Learning Algorithm for Boltzmann Machines, Cognitive Sci., Vol. 9, pp. 147-169 (1985).
15) 倉田耕治：神経回路モデルとしてのボルツマン・マシン，数理科学，Vol. 25, No. 7, pp. 23-28 (1987)．
16) Rumelhart, D. E.: Feature Discovery by Competitive Learning, Cognitive Sci., Vol. 9, pp. 75-112 (1985).
17) Kohonen, T.: A Simple paradigm for the self-organized formation of structured feature maps. In Competition and Cooperation in Neural Nets (Amari, S. and Arbib, M. A. ed.), Lecture Notes in Biomathematics 45, Springer-Verlag (1982).
18) 麻生英樹：ニューラルネットワーク情報処理—コネクショニズム入門，あるいは柔らかな記号に向けて—，産業図書 (1988)．
19) Marr, D.: Vision—A Computational Investigation into the Human Representation and Processing of Visual Information, W. H. Freeman (1982) (乾 敏郎，安藤広志訳：ビジョン—視覚の計算理論と脳内表現—，産業図書 (1987))．
20) 稲葉則夫：ニューラル・ネット製品の第一弾出そろう，日経エレクトロニクス，1987. 8. 10 (No. 427)，pp. 67-124 (1987)．
21) Miyata, Y.: SunNet Version 5.2: A Tool for Constructing, Running, and Looking into a PDP Network in a Sun Graphic Window, ICS Report 8708, Institute for Cognitive Sci., Univ. of Califolnia at San Diego (1987).

(昭和 63 年 7 月 13 日受付)