設計自動化 59－12
（1991. 10. 9）

ボルツマンマシンの耐故障性について

岡田 勲　　当麻 喜弘

東京工業大学 情報工学科

あらまし　現在のニューラルネットワークの研究は、特徴や性能の評価にシミュレーションを用いる場合が多いが、将来においてはネットワークをハードウェア上で実現することが必要となってくる。そして、その場合には素子数が膨大になり、これを実現するためにはある程度の素子パラメータの変動が生じたり、故障も多く起こり得るであろう。本論文では、特にハードウェアモデルとしてボルツマンマシンを用い、それに巡回セールスマン問題を適用して、ハードウェア化に伴い生じる問題であるユニットの出力固定故障・しきい値の誤差・結合の断線や誤差による影響についての検討をシミュレーションで行ない、その問題点を明らかにした。

Fault Tolerance of Boltzmann Machines

Isao OKADA　Yoshihiro TOHMA

Department of Computer Science, Tokyo Institute of Technology

2-12-1, Ookayama, Meguro-ku, Tokyo, 152, Japan

Abstract　This paper describes the fault tolerance of Boltzmann Machines. Neural networks are studied mainly by simulations, but in the near future it will be implemented in hardware chips. Since the hardware implementation may have many components and connections, the accuracy of implementation parameters may be one of serious problems and further the stuck-at-faults may be encountered in many components. Therefore, it is important to reveal the capacity of fault as well as error tolerance in neural networks. We examined the effect of stuck-at-faults, disconnections, and errors in biases and connections, applying Boltzmann Machine to the traveling salesman problem.

1　序論

人間は、「あいまいさ」を持つ情報を巧みに処理する。「文字を読む」とか「言葉を理解する」といったことを人間は何気なく行なっているが、このような処理は従来のフォンノイマン型コンピュータにとっては困難であることがわかってきた。そこで、新しい超並列処理計算機アーキテクチャの研究が重要になってきた。その一例として、人間の脳の情報処理を研究の対象として、その機構や原理などを生物学的ではなく情報処理の立場から解明しようとしているのが、ニューラルネットワークの研究であり、ニューロコンピューティングでもある。

現在のニューラルネットワークの研究は、シミュレーションでその特徴や性能を評価している一方で、ハードウェア化の試みが行なわれている段階である。しかし、将来ニューラルネットワークの理論や原理がある程度確立された段階においては、大規模なネットワークを構築し実用レベルの実験を行なうことも必要になってくる。その段階ではネットワークをハードウェア上で実現することも必要になってくる。そして、その場合にはユニット数やユニット間の接続が膨大になり、これを実現するためにはある程度の素子パラメータの変動も生じるであろうし、故障も多く起こり得るであろう。例えば、ユニット間の信号にアナログの電圧値を用いたり、光の強度を用いたりした場合はその変動の影響を受けるであろうし、ユニットの数が多くなればユニットの故障も無視は出来なくなる。そこで、問題となるであろう要因である、ユニット間の結合の断線と重みの誤差、ユニットのしきい値の誤差と出力値の故障によるニューラルネットワークの性能への影響が心配される。

以上のことを背景として、本研究では、ニューラルネットワークのハードウェア化に伴い生じる問題である、ユニットの故障や重み・しきい値の誤差そして結合の断線による影響について検討を行なった。本論文では、ハードウェアモデルとしてボルツマンマシンを用い、それにセールスマン問題を適用してシミュレーションを行ない、その評価や検討結果を述べる。

2　ボルツマンマシン

本節では、相互結合型ニューラルネットワークの一つであるボルツマンマシンについて説明する。そして、組合せ最適化問題である巡回セールスマン問題への適用方法について説明する。

2.1　ボルツマンマシンの構造

図1に示したものがボルツマンマシンのアーキテクチャである。

円で表されものは**ユニット**と呼ばれる脳のニューロンに相当するもので、多入力・一出力の比較的単純な処理を行ない、一般的には入力の総和によって出力 u_i が決定される。各々のユニットは、出力を決定するパラメータとしての**しきい値** θ_i を持つ。

また、各ユニットは**結合**と呼ばれる自分以外のユニットへのフィードバック線を持っている。それぞれの結合に

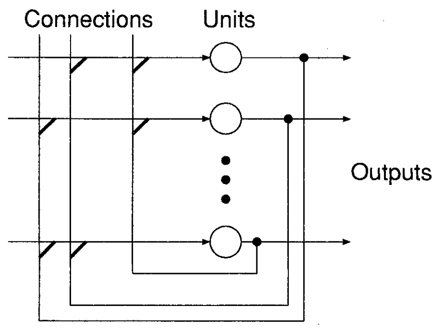

図 1: ボルツマンマシンのアーキテクチャ

は**重み**と呼ばれる量が付随しており、i 番目のユニットから j番目のユニットへの結合の重みを w_{ij} と記す。この結合の重みは**対称**となっている。つまり、

$$w_{ij} = w_{ji} \tag{1}$$

である。また、自分自身へのフィードバック線は持っておらず、

$$w_{ii} = 0 \quad for\ all\ i \tag{2}$$

となる。

ネットワークのすべてのユニットの出力を、そのネットワークのその時点での**状態**といい、$\boldsymbol{\alpha}$ や $\boldsymbol{\beta}$ などのギリシャ文字で記すことにする。ユニット数が n の場合、状態は n 次元ベクトルで表現される。

2.2　ユニットの出力値決定法

ボルツマンマシンは状態を遷移させていくことにより、安定な状態へと収束していくのである。ここでは状態を遷移させる方法、つまりユニット出力値の決定方法について説明する。

i 番目のユニットの入力の総和 $i_i(\boldsymbol{\alpha})$ は式 (3) で表される。

$$i_i(\boldsymbol{\alpha}) = \sum_j w_{ji}u_j(\boldsymbol{\alpha}) + \theta_i \tag{3}$$

ボルツマンマシンで用いられるユニットの出力は 1 もしくは 0 の 2 値をとり、しかも入力値に応じて確率的に決定される。つまり、i 番目のユニットの出力 u_i はその入力 i_i に応じて、確率

$$P(u_i = 1) = \frac{1}{1 + \exp(-i_i/T)} \tag{4}$$

で値 1 をとり、確率

$$P(u_i = 0) = 1 - P(u_i = 1) \tag{5}$$

で値 0 をとるものとする。

式 (4) の T は**ネットワークの温度**と呼ばれる値で、この T を変化させることにより、入力 i_i と確率 $P(u_i = 1)$ との関係は図 2のように変化する。

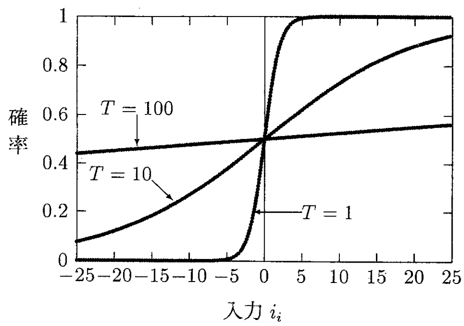

図 2: 入力値と確率の関係

T を変化させることはネットワークの揺らぎを変化させることに相当する。T が大きい (温度が高い) と式は i_i の値に鈍感となり、ほぼ任意に出力値が決定される。また、T が小さい (温度が低い) と式は i_i の値に対して敏感になり、$T \to 0$ の極限では Hopfield 型ニューラルネットワークと同様に決定的なモデルとなり、揺らぎがなくなる。この性質を利用することによって、ボルツマンマシンは最適状態へとうまく遷移するのである。

2.3 ボルツマンマシンの動作手順

以上の準備のもとに、ボルツマンマシンの動作手順をプログラムで書き表すと図 3のようになる。

```
1:  program Boltzmann Machine
2:  begin
3:      for i := 1 to ユニット数 do
4:          ユニット u_iに初期値を設定 ;
5:      T := T_start ;
6:      t := 0;
7:      repeat
8:          for i := 1 to ユニット数 do begin
9:              すべてのユニットの中からランダムに
                1 つのユニット u_iを選ぶ。;
10:             確率 P(u_i = 1) に従い、
                ユニットの出力を 1 にする。;
11:         end ;
12:         t := t + 1 ;
13:         T := T(t) ;
14:     until T = T_end ;
15: end
```

図 3: ボルツマンマシンの動作手順

まず、すべてのユニットに 0 もしくは 1 の初期値を設定する。つぎに、一定の温度で 8 行目から１１行目までの処理を繰り返す。この処理では、最初にすべてのユニットから出力値の変更を試みるユニットをランダムに選ぶ。そして、その選ばれたユニットの入力値から式 (4) (5) によって与えられる確率にしたがって選ばれたユニットの出力値を変更する。この処理をほぼ全ユニットで行なってから、アニーリングのスケジュールと呼ばれる関数 $T(t)$ を用いて温度 T を少しだけ下げて、その温度が終了温度でなければ更に 8 から１１行目までの処理を繰り返すのである。

このように状態遷移は**非同期**で、ユニットの出力値を変えることによってネットワークの状態が変化していく時、唯 1 つのユニットのみがその出力値を変化させるのである。

この動作に従い状態を遷移させていくと、ボルツマンマシンは次に定義される**エネルギー**が小さくなる状態へと遷移する。

定義 1 (エネルギー) 対称結合が成立し動作が非同期の場合には、次に示すエネルギーと呼ばれる状態関数が定義できる。

エネルギーとはネットワークにおける結合の活性度を表しており、ネットワークの状態が決まると式 (6) によって定まる量のことである。

$$E(\boldsymbol{\alpha}) = -\frac{1}{2}\sum_i\sum_j w_{ij}u_i(\boldsymbol{\alpha})u_j(\boldsymbol{\alpha}) + \sum_i \theta_i u_i(\boldsymbol{\alpha}) \tag{6}$$

エネルギー値が小さいほどネットワークは活性化している。 □

ネットワークを収束させようとする状態において、このエネルギー関数は最小値となるが、この関数には極小値が多数存在する。そのため状態遷移が決定的である時には、極小値をとる極小状態に捕まってしまえばそこから抜け出せなくなる危険性がある。しかしボルツマンマシンの状態遷移は確率で決定されるので、極小状態から抜け出すことができるのである。これがボルツマンマシンの特徴である。

しかし、状態遷移が確率的であるために最適状態から抜け出す確率も少なからず存在する。そのため、うまく安定な状態へと収束しなくなる恐れが生じる。そこで、状態遷移を行ないながらネットワークの温度を低くしていき、遷移の方法を徐々に決定的にしていくのである。このことによって、状態遷移の最終段階において収束する状態がある程度決まったら、大幅なエネルギーの変動がある状態遷移は生じないようにして、最適状態から抜け出さないようにしているのである。

この方法は、シミュレーテッドアニーリングと呼ばれている。これは、焼きなましを計算機上で模擬することであり、温度というゆらぎを系に与えることにより、エネルギーの極小状態に捕らわれることなく、最適状態を得るための手段である。これは、アニーリングのスケジュール $T(t)$ を用いて温度 T をうまく制御することによって実現される。

2.4 巡回セールスマン問題への適用

2.4.1 巡回セールスマン問題

本論文で適用する最適化問題である巡回セールスマン問題について具体的に説明する。

巡回セールスマン問題は、一人のセールスマンが N 都市を一度ずつ訪問し、出発した都市に再び戻るような経路のうちで、最短のものを求める問題である。

巡回セールスマン問題では、$N \times N$個のユニットを用いて、「行を都市の種類に対応させ、列を訪問する順番に対

応させる」ようにユニット毎に意味を割り当てる。例えば、u_{ik} の出力が 1 であることは、都市 i を k番目に訪問する事を意味する。セールスマン問題のこの表現形式では図 4 のような状態の時に有効な解が求まるのである。

	1	2	3	4	5
A	□	□	□	□	■
B	■	□	□	□	□
C	□	□	■	□	□
D	□	■	□	□	□
E	□	□	□	■	□

図 4: ユニットの構成

図 4では、道順は次のような巡回ルートになる。

$$B \rightarrow D \rightarrow C \rightarrow E \rightarrow A$$

各ユニット毎に明白な意味が存在するという意味では局所的な情報表現であるが、セールスマン問題では「巡回のルートを選択できる」ために起点が決まっておらず、全体としてはある意味では分散的な情報表現が行なわれていると言える。

そして、その全体としての分散的情報表現のために、セールスマン問題の情報表現形式はある程度の耐故障性を持った表現であると言える。即ち、セールスマン問題の表現形式は耐故障性の評価を考えた場合には、比較的良い表現形式であり、また逆に、耐故障性のある情報表現が行なえる比較的良い最適化問題であるとも言える。そのため、本論文では ボルツマンマシン上に適用する最適化問題をセールスマン問題として故障や誤差の評価を行なうことにした。

2.4.2 エネルギー関数の構成法

まず、セールスマン問題の制約条件「都市を必ずただ一度だけ訪問しなければならない」を満たすための制約関数を式 (7) のようにする。

$$E_1 = \sum_i (\sum_k u_{ik} - 1)^2 + \sum_k (\sum_i u_{ik} - 1)^2 \qquad (7)$$

さらに、セールスマン問題の目的「都市の経路の最短のもの」を満たすための目的関数を式 (8) のようにする。なお、d_{ij}は都市 i と都市 j との距離とする。

$$E_2 = \sum_i \sum_j \sum_k d_{ij} u_{ik} (u_{jk+1} + u_{jk-1}) \qquad (8)$$

式 (7) , 式 (8) のエネルギー関数と制約係数 A、目的係数 B より、ネットワーク全体のエネルギー関数 E を式 (9) のように構成する。

$$E = \frac{1}{2}(AE_1 + BE_2) \qquad (9)$$

先に定義した一般的なエネルギーの定義式(6) と比較することにより、結合の重みとしきい値を求めるのである。

3 シミュレーション方法

本節では、故障や誤差による影響の評価を行なうためのシミュレーション方法について説明する。

3.1 故障と誤差

ユニットの故障としては、出力が入力に関係なく固定値 0 又は 1 を出力する故障を考え、結合の故障としては、重みが 0 となる断線を考える。

本論文で扱う故障のモデルを次のようにする。

- ユニットの出力が固定値 0 を出力する故障
- ユニットの出力が固定値 1 を出力する故障
- 制約関数側の結合が断線する故障
- 目的関数側の結合が断線する故障

しきい値・結合の重みの値が、素子の特性やバラツキにより本来設定すべき値 x より外れる事を誤差という。誤差の設定には、誤差の成分 G のパラメータがあり、式 (10) の誤差挿入関数 $X(x, G)$ を使用する。$rand()$ を -1 から 1 までの一様乱数を返す関数とする。

$$X(x, G) = x \times (1 + G \times rand()) \qquad (10)$$

結合の重みの誤差には、ボルツマンマシンの前提条件である対称結合の条件である式 (1) が成立しないような誤差 (**非対称誤差**) と、その反対に等価性が成立するような誤差 (**対称誤差**) が考えられる。

非対称誤差が存在する場合は、先のエネルギーの概念が成立しなくなり、収束する保証がなくなってしまう。すなわち、最悪の場合は発振などの状況に陥り解が読み取り不能になってしまう。非対称誤差が発生した場合のニューラルネットワークのエネルギーについては、数学的立場から再度検討する必要があると思われるので、本論文では取り扱わない。断線の場合もこの対称性を考慮している。

本論文で扱う誤差のモデルを次のようにする。

- ユニットのしきい値の誤差
- 制約関数側の結合の重みの対称誤差
- 目的関数側の結合の重みの対称誤差

3.2 評価方法

1 回の試行で得られた状態は、最適化問題の制約条件を満たさず無意味な**無効回答**と、制約条件を満たし最小値 (最適値) ではないにしても解として意味のある**有効回答**に分類される。そして、この有効回答の中に**最適解** が含まれるのである。

ニューラルネットワークの評価は、各パラメータを設定した後に指定された試行回数分のシミュレーションを行ない、得られた有効回答率・最適解数・道のり差を基準として行なうこととする。ただし、有効回答率・最適解率・道のり差はそれぞれ式 (11)・(12)・(13) で与えられる値である。

有効回答率

1 回の評価で問題の制約条件を満たした有効な解答を得ることが出来た割合を**有効回答率**という。

$$有効回答率 = \frac{有効回答数}{試行回数} \qquad (11)$$

最適解率

１回の評価で問題の最適解を得ることが出来た割合を**最適解率**という。

$$最適解率 = \frac{最適解数}{試行回数} \tag{12}$$

道のり差

道のりの平均から問題の最適解の道のりを引いたものを**道のり差**という。なお、最適解は、すべての道順を調べた結果を用いている。

$$道のり差 = \frac{\sum 有効回答の道のり}{有効回答数} - 最適解の道のり \tag{13}$$

4 故障と誤差の評価

本節では、ボルツマンマシン上でユニットの故障、しきい値・結合の重みの誤差、及び、結合の断線について述べる。ただし、本論文では組合せ最適化問題である５都市巡回セールスマン問題を適用した場合に限定して実験を行なった。

実験に用いるデータは、付録の図10の５つのデータとする。制約係数 A と目的係数 B の比率はそれぞれ付録の表1に示す。アニーリングのスケジュール $T(t)$ は、

$$T(t) = \frac{T_{start}}{1 + t/\tau} \tag{14}$$

を用いる。τ は時定数とよばれるパラメータである。

１回の試行における解の選択方法として、T が最終温度になったらその温度を保って1000回状態遷移行ない、その中でもっとも出現確率の高いものを解とする方法を用いた。

各パラメータは、初期温度 $T_{start} = 500$ 、終了温度 $T_{end} = 0.8$ 、時定数$\tau = 10$ とする。

以下に示すシミュレーション結果は、図10に示した５つのデータにおいて試行をそれぞれ100回ずつ行なった結果の平均である。

4.1 故障による影響

固定故障の実験は、正常時と同じ条件のもとで選ばれたユニットに０または１固定故障を挿入して行なった。故障させるユニットはネットワーク上からランダムに選び、状態遷移を試みる間はそのユニットは必ず１または０の値を出力しているものとする。

断線故障の実験は、断線させる結合の重みの値を０にして行なった。選ばれる結合は、ネットワーク上からランダムに選び出すものとする。

4.1.1 0 固定故障の影響

０固定故障を挿入した時の影響について調べた結果を図5に示す。

故障が１つのユニットのみで発生している場合は、有効回答・最適解・道のり差のすべてにおいてほとんど故障の影響が現れなかったといえる。５都市巡回セールスマン問題を解くために構成されたネットワークでは最適状態が１０状態存在する。もし１つのユニットが０固定故障を起こすとすると、最適状態は２つなくなるが依然として８状態残っており、最適状態は存在する。ボルツマンマシンはエネルギー値の小さな状態を探索しながら収束していく性質があるので、０固定故障が１ユニットで発生した時はその影響が現れなかったのであろう。

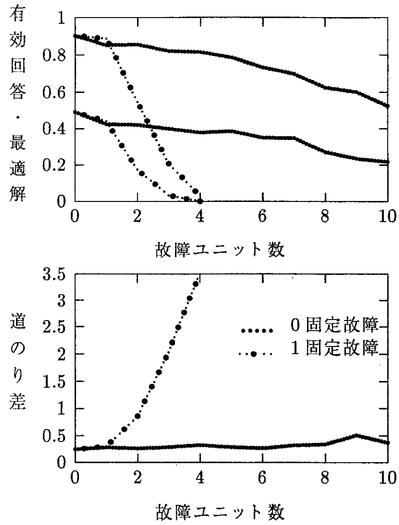

図 5: 固定故障における影響

故障が複数のユニットで発生している時は、その故障の増加にともない性能は低下しているが、それは急激なものではなく緩やかである。これは巡回セールスマン問題における出力０のユニットは大きな意味を持たないからであるといえる。

4.1.2 1 固定故障の影響

１固定故障を挿入した時の影響についてシミュレーションした結果を図5に示す。

１固定故障が１ユニットで発生した場合は、ほとんどその影響は現れなかったが、故障ユニット数が２ユニット以上になるとその影響は顕著に現れ、性能が著しく低下した。

故障ユニットが１ユニットの時は最適状態が依然として存在するので、この条件下のボルツマンマシンは最適解に収束することが可能である。従って、１固定故障が１ユニットのみで発生した場合にはそれほど影響が現れなかったのだと考えられる。

しかし、故障ユニットが２ユニット以上になった時には制約条件を満たすような状態になることが不可能な場合が存在し、最適状態になることができない故障が発生するので、極端に性能が落ちたのだと考えられる。

4.1.3 固定故障への対応策

以上のことより、実際にハードウェアを構築する場合は次に示す注意点がある。

留意点 1 (固定故障への対応)

ユニットにおいて固定故障が起こった時には大きな意味を持たない出力の方向へ誤るような素子を、ユニットと

して用いるべきである。 □

このシミュレーションでは巡回セールスマン問題を用いてネットワークを構成したが、これはユニットの出力が 1 の時に意味を持つ構成であった。そのため、0 固定故障よりも 1 固定故障の方がその影響が顕著に現れたと考える。

4.1.4 制約関数側の結合の断線による影響

制約関数を構成する結合を 1 本ずつ切りとることによって実験を行ない、その影響を調べた。

シミュレーション結果を図 6に示す。

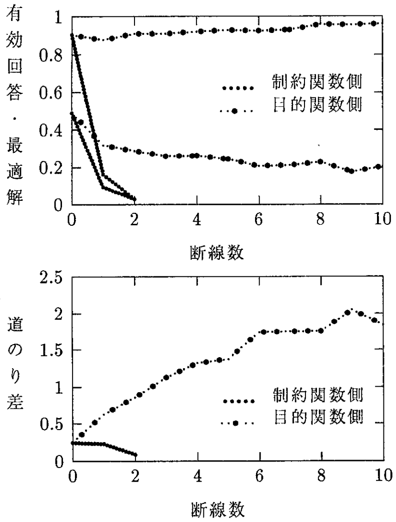

図 6: 結合の断線による影響

制約関数側の結合を断線することによる影響は、有効回答・最適解において顕著に現れている。道のり差には、断線無しと断線 1 本有りとではほとんどその差はないが、これは有効回答中に最適解が占める割合が変化しなかったためである。

ボルツマンマシンで巡回セールスマン問題を表現するためには、必ず制約条件を満たし解が読みとれるようにしなければならなかった。しかし、制約条件を満たす状態へと収束するように決定された結合が断線したために、ユニットの出力値に狂いが生じ、制約条件を満たすような状態へとは遷移しなくなったため、このように悪影響が現れたと考えられる。

留意点 2 (制約関数側の結合の断線)
制約関数側の結合が断線すると、ボルツマンマシンの性能は大幅に下がってしまうので、ユニット間の結合は断線しないような構造にすべきである。 □

4.1.5 目的関数側の結合の断線による影響

目的関数を構成する結合を 1 本ずつ切りとることによって実験を行ない、その影響を調べた。

シミュレーション結果を図 6に示す。

目的関数側の結合の断線による影響は、最適解・道のり差において現れた。

制約関数側の重みは正常であるので、ネットワークは制約条件を満たす状態へと遷移していく。従って、有効回答が断線によって影響を受けなかったと考えられる。

断線の本数を増加することによる影響は 5 本あたりまで急激な変化が見受けられたが、それ以降は緩やかな変化にかわっている。断線数が 5 本以上になるとより小さな極小解へと向かうことには鈍くなり、制約条件を満たすように状態が遷移していくためであると思われる。

留意点 3 (目的関数側の結合の断線)
目的関数側の結合の断線は多量にならないように注意すべきである。 □

4.2 誤差による影響

誤差はすべての値に挿入されるものとし、その値は誤差関数 (10) によって計算される。

4.2.1 しきい値の誤差による影響

しきい値の誤差 G_{th} を 0% から 50% まで 5% 刻に変化させた場合について実験を行ない、その影響を調べた。

シミュレーション結果を図 7に示す。

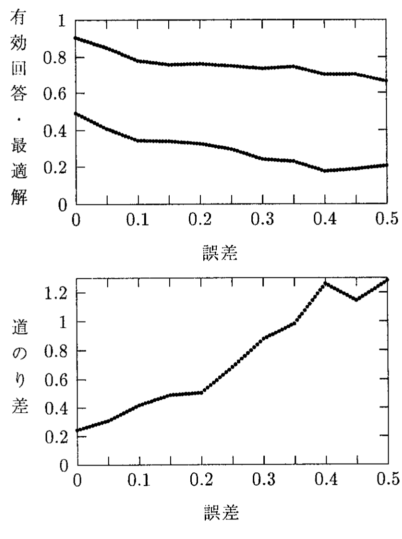

図 7: しきい値の誤差による影響

しきい値の誤差による影響は、有効回答・最適解・道のり差のすべてに明らかに現れている。特に道のり差は 6 倍になり、その影響が顕著に現れている。

しきい値はユニットの出力を決定する上で非常に重要な役割を担っていた。よって、これに誤差が生じることによって、ネットワーク全体に大幅な狂いが生じ、正確な動作が望めなくなってしまう。従って、大幅な悪影響が現れたと考えられる。

留意点 4 (しきい値の誤差)
しきい値の誤差は有効回答・最適解・道のり差のすべてに悪影響を及ぼすので、その精度には十分注意を払うべきである。 □

4.2.2 制約関数側の重みの誤差による影響

制約関数側の重みの誤差 G_{res} を 0% から 50% まで 5% 刻に変化させた場合について実験を行ない、その影響を調べた。

シミュレーション結果を図8に示す。

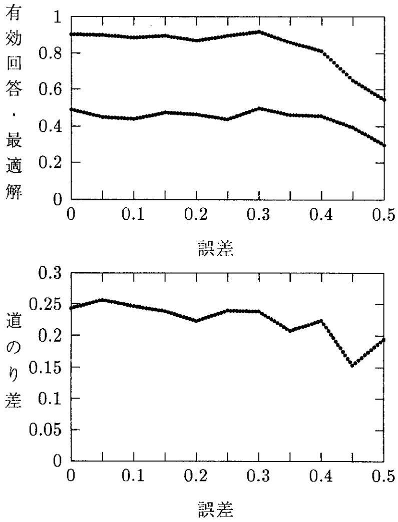

図 8: 制約関数側の重みの誤差による影響

制約関数側の重みの誤差による影響は、誤差 30% 以上になって有効回答・最適解に現れたが、道のり差は逆に誤差が大きくなるにつれて結果が良くなってきた。

誤差が 30% となるまで有効回答・最適解に影響が現れなかったのは、制約関数側の係数を十分に大きくとっていたためであると思われる。その影響について有効回答と最適解を比べると、誤差 30% を越えてからの減少は有効回答の方が急峻である。これは、制約関数側の重みの誤差によって最初に影響を受けるのはエネルギー値が大きな状態、つまり不安定な状態からであるためと思われる。不安定な状態であれば隣接した極小状態とのエネルギー壁が低いため、大きな誤差が生じた時には影響が及ぼされやすいからである。

誤差が大きくなるにつれて道のり差が改善されたのは、有効回答中の最適解が占める割合が大きくなったためだと思われる。

留意点 5 (制約関数側の重みの誤差)
制約関数側の重みの誤差が大きくなるとその影響は顕著に現れるが、少量のずれに対してはその影響は覆い隠すことが可能であるので、制約関数側の重みは大幅にずれないように注意しさえすれば、さほど心配する必要はない。 □

4.2.3 目的関数側の重みの誤差による影響

目的関数側の重みの誤差 G_{pur} を 0% から 50% まで 5% 刻に変化させた場合について実験を行ない、その影響を調べた。

シミュレーション結果を図9に示す。

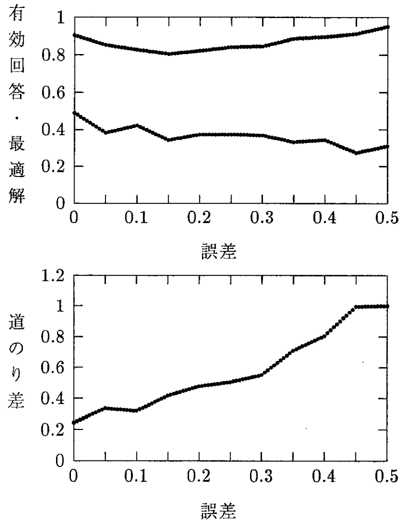

図 9: 目的関数の重みの誤差による影響

目的関数側の重みの誤差による影響は、有効回答へはあまり現れていないが道のり差には顕著に現れていた。

目的関数側の重みに誤差が生じるとエネルギー曲面が変形し、状態間の関係が変わってしまう。例えば、本来エネルギー値の小さな状態が結合の誤差によってエネルギー値が大きくなり、代わってエネルギー値の大きかった状態が小さくなり、状態間の関係が逆転する恐れがある。道のり差に誤差の影響が現れたのは、このことが原因であると思われる。

有効回答にさほど影響が現れなかったのは、目的関数側の重みのみに誤差が生じたためであると考える。

留意点 6 (目的関数側の重みの誤差)
目的関数側の重みの誤差は、その最適解に影響を与えるのでその精度に注意を払うべきである。 □

5 結論

本研究では、ボルツマンマシンのハードウェア化の際に生じるであろう誤差や故障の影響について明らかにした。

5.1 故障と誤差への対応

ボルツマンマシンのハードウェア化に際して生じる問題である故障と誤差に対して、ボルツマンマシン上で 5 都市巡回セールスマン問題を適用した場合に限定してその対応策を明らかにした。

故障と誤差への対応策は次に挙げるものである。

(1) 1 固定故障が 1 ユニットのみで発生しているのなら、ボルツマンマシンの性能に影響を与えないが、多ユニットになる場合は大幅にその性能は低下するので、固定故障が発生した場合にはボルツマンマシンに影響を与えないために、0 出力方向に誤るようなユニットにするべきである。

(2) しきい値の誤差による性能への影響は大きいので、

その精度には十分注意すべきである。

(3) 結合の重みの誤差に関しては、誤差が大きくならないようにすればある程度ボルツマンマシンの性能は維持できるが、制約関数側の結合の断線によってその性能は悪化するので、誤差は生じても断線しないような結合にすべきである。

5.2 今後の課題

本論文ではボルツマンマシンの故障と誤差の影響について調べたが、ボルツマンマシン上に巡回セールスマン問題を適用した場合のみにおいて行なった。しかし、他の最適化問題を適用した場合や学習の際の故障や誤差についても実験を行ない、その影響について調べる必要がある。

5.2.1 耐故障性のある情報表現

固定故障による影響は、0 固定故障ではその数が多くならない限りさほど現れなかったが、1 固定故障では複数ユニットに及ぶと大幅に現れ、ボルツマンマシンの性能は極端に悪化した。これは、巡回セールスマン問題を適用したボルツマンマシンの解の表現方法に問題があると思われる。1 を出力するユニットと 0 を出力するユニットの意味するところに大きな違いがあり、出力 1 の方が強い意味を持つ表現方法を用いているので、固定故障の場合には大きな差となって現れたと考えられる。

そこで、出力が 1 であるユニットと 0 であるユニットの持つ意味が等価になるような情報表現方法を考えなければならない。

また冗長なユニットを用いることによって、フォールトトレラント化を計ることができものと思われる。

耐故障性のある情報表現については詳しく検討する必要がある。

5.2.2 故障や誤差による学習の影響

本論文では、ユニットのしきい値や結合の重みがすでに決定されているボルツマンマシンの評価を行なったが、ボルツマンマシンはしきい値や重みを学習によって得るという機能を持っている。そこで、学習時における故障や誤差の影響について調べることも興味深い研究課題である。

参考文献

[1] 麻生 英樹. ニューラルネットワーク情報処理. 産業図書,1988.

[2] Emile Aarts and Jan Korst. *Simulated Annealing and Boltzmann Machines.* JOHN WILEY and SONS, 1988.

[3] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1984.

[4] G.Hinton, T.Sejnowski, and D.Ackley. Boltzmann machines: constraint satisfaction networks that learn. *Tech. Rep. CMU-CS-84-119, Carnegie-Mellon Univ.*, 1984.

[5] J.J.Hopfield and D.W.Tank. Neural computation of decisions in optimization problem. *Biol.Cybern*, 1985.

A 付録

地図データ	制約係数：目的係数
Map1	12:1
Map2	12.2:1
Map3	15.6:1
Map4	16.2:1
Map5	13.3:1

表 1: 各地図データの制約係数と目的係数の比率

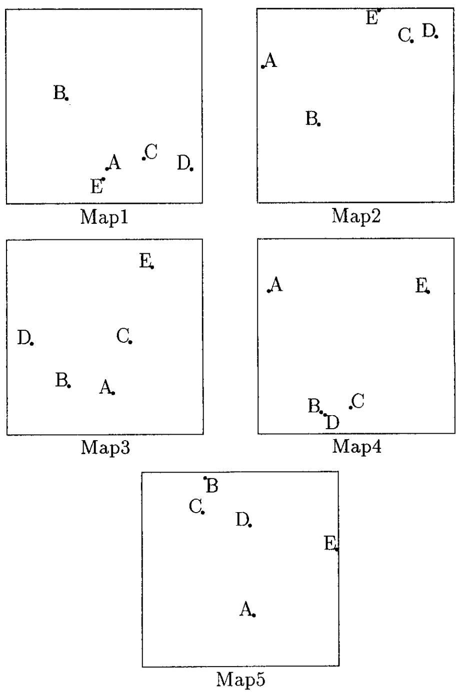

図 10: 地図データ