特別研究報告書

パルス駆動型ニューラルネットにおける順序回路

指導教官　　富田 眞治 教授

京都大学工学部情報学科

津田 晃寿

平成 13 年 2 月 13 日

パルス駆動型ニューラルネットにおける順序回路

津田 晃寿

内容梗概

人間の脳のように知的なシステムを構築することが我々の最終的な目的である。

我々は、そのようなシステムを構築する際に必要となると考えられる、神経網のシミュレータを作成した。そのため、シミュレートする神経網の模倣品を作る必要があるが、それは遺伝的アルゴリズム (GA) によって生成することにした。

また、我々は脳において記憶こそがその本質であると考える。しかし、脳の記憶の構造は、よくわかっていない部分が多い。そこで我々は、その記憶の構造を知ることを当面の目標とした。

このような目標から、本シミュレータで採用する神経細胞のモデルは、実際の神経細胞に近いモデルであることが求められる。よって、よく知られる単純な形式ニューロンモデルではなく、入力パルスの蓄積によって発火する、パルス駆動型ニューロンモデルを採用している。さらに我々は、このモデルをより実際の神経細胞に近づけるため、絶対不応期と相対不応期の概念を導入している。

一方、記憶の構造を探るという目的のために、GA によって生成されるネットワークは記憶のための構造を持ったものであって欲しい。そこで「簡単な迷路ならば解くことができるネットワーク」を考える。これは、迷路を解くためにはどちらから来たかという短期記憶や迷路の解き方という長期記憶が必要となるためである。また、このような具体的な目的を持ったネットワークならば、論理的に構成できるので、GA の初期個体として構成しやすい。

この初期個体を作成するために、まず簡単な迷路を解くためのアルゴリズムを求めた。その際、記憶の構造を知るという目的から、アルゴリズムは迷路を解く際に記憶を用いるものとする。

次に、それを論理順序回路として構成し、ネットワーク上に実装した。

その際、順序回路をネットワーク上に実装する方式として、ニューロンが論理素子の振舞いをするモデルを与えるアプローチをとった。そのために、ニューロンによって論理素子を実現する際に、まず入出力特性によって分類できる様々なモデルを示した。

また、本研究では簡単のため、順序回路には入力を 2 値的に与えた。そのた

めニューロンモデルは、簡単に実現できる、入出力特性が線形のものを採用した。そのようなモデルとして、例えば AND の振る舞いをするニューロンであれば、全ての入力枝にパルスが入った時のみ発火するものを作成した。

一方今回、回路構成において、迷路を解く際に必要な記憶のデータ構造がLIFOで与えられるため、順序回路に与える記憶の構造はスタックとした。

また、順序回路の挙動はクロックによって制御される。ただし、ここでのクロックは、周囲の壁の状況が変化した時に変化する。そして、周囲の壁の状況が変化した時に、クロックの立ち上がりによって状態が更新される。

さらに、入出力特性が閾値を持つニューロンのモデルを作成する方法を提案した。前述のように作成したネットワークにおいて、論理素子の振舞いをするニューロンが線形の入出力特性を持つということは、入力によっては論理値の 0 と 1 を明確に分けることができない場合が存在する。しかし、実際の神経細胞の入出力特性は閾値があるので、どんな場合でも論理値の 0 と 1 を明確に分けることができる。そこで、より実際の脳に近いシミュレーションを行うためには、論理素子の振舞いをするモデルの入出力特性が閾値を持った形であればよい。

この提案に基づいて、入出力特性が閾値を持つ形の論理素子の振舞いをするモデルを作成できれば、後は今回作成した順序回路を形成している線形の入出力特性を持つモデルと交換するだけで、より実際の脳に近いシミュレーションができる初期個体を生成できると考えられる。

Sequential Circuits on a Pulse-Driven Neural Network

Akihisa TSUDA

Abstract

Our last purpose is to develop a really intelligent system like human brain.

We made a simulator of neural network. We need it when we develop such a system. We need to build an imitation of brain to simulate. We build it with the Genetic Algorithm (GA).

Memory is the most important matter in the brain. But we scarcely know its structure. Our present purpose is to know it.

This purpose demands that this simulator adopts the neuron model just like real neurons. Therefore, we do not adopt the well-known simple neuron model. We adopt the pulse-driven neuron model. This neuron model fires when input pulses accumulate enough. Furthermore, we introduce the conception about absolute refractory period and relative into this model like real neurons.

On the other hand, in order to know the structure of brain's memory, we need the network with the structure of memory when we build it with GA. Therefore, we develop a network which can solve easy mazes. Such the network requires short term memory and long term memory. This short term memory indicates which direction the agent moving in a maze comes from. This long term memory indicates how to solve mazes. This network has the concrete purpose that it solves mazes. We can build such the network logically. So this network makes it easy to build the primary individual of GA.

To build this primary individual, firstly, I presented the algorithm to solve easy mazes. In order to know the structure of memory, this algorithm should have memory to solve mazes.

Secondly, I reduced the algorithm to a logical sequential circuit and implemented it on the network.

Then, in order to implement this circuit on a network, I took the approach that I made a model which behaves as a logical device. Therefore, firstly, I presented the several models classified in the input-output peculiarity. Secondly, I reduced neurons to logical devices.

In this paper, simply, I gave this circuit inputs like two logical values. There-

fore, I adopted the neuron model with the linear input-output peculiarity. We realize this model easily. In this model, for example, in order to make the neuron which behaves as a logical device *AND*, I made the neuron which fires only when pulses come to all input lines.

In the development of this circuit to solve easy mazes, the data structure of memory is LIFO. Therefore, I gave the circuit the structure of memory as a stack.

On the other hand, the clock controls the behavior of this circuit. In this paper, the clock's value changes when the inputs of walls around the agent moving in a maze changes. And then, the change of the clock's logical value from 0 to 1 updates the state of the circuit.

And then, I presented the method to make the model whose input-output peculiarity has threshold. In the network which I built in this research, the neurons behaving as logical devices has the linear input-output peculiarity. This suggests that it is difficult to distinguish between the logical value 0 and 1 clearly when some inputs come. But a real neuron's input-output peculiarity has threshold. This suggests that it is always possible to distinguish. Therefore, in order to simulate like a real brain, the input-output peculiarity of the model behaving as logical device should have threshold.

We will have made the model which behaves as logical device and has the input-output peculiarity with threshold. Only exchanging the neuron model with linear input-output peculiarity, we will be able to build the primary individual to simulate like a real brain.

パルス駆動型ニューラルネットにおける順序回路

目次

第 1 章　はじめに

人間の脳のような知的なシステムを作ることが我々の研究の動機であり、最終的な目標である。

我々の言う人間のように知的なシステムとは、状況を理解し、常識を有し、言葉を理解する。よって、このシステムは様々な知的システムに応用できるだろう。例えばそれは、有能なアシスタントになり得るかもしれない。

そのようなシステムを作ることを考えた時、従来あるアプローチとしては人工知能研究がまず最初に思い付くだろう。しかし、この方法では人間の論理的な思考過程を模倣することはできても、人間自身が説明できない思考過程を模倣することは困難である。よって従来の人工知能の方法の延長線上に、脳のような知的なものができるとは考えにくく、また、できるとしてもアプローチとして遠いと思われる。

そこで、我々はそのようなシステムを作る上で必要になると考えられる神経網のシミュレーションを行うためのシミュレータを作成した。よって、シミュレートする脳の神経網の模倣品を構築する必要がある。しかし、我々は脳の神経網のようなものを作り方を知らない。そこで、それは GA によって生成することにする。

また我々は、脳において重要な役目を果たしている記憶の構造を知ることを当面の目標とした。そのために、脳の記憶の構造をシミュレートできるようなネットワークを生成する必要がある。

そのため、本研究ではそれを GA によって生成するために必要な初期個体を、パルス駆動型ニューラルネットワーク上に順序回路として構成することが目的である。

本論ではこの目的のために、まず本研究の位置付けを明確にした上で、上述のネットワークを作成するために必要な、パルス駆動型ニューラルネットワークにおける順序回路の実現方法について述べる。以下、第 2 章では本研究の背景と我々が採用するニューロンモデルについて述べる。次に第 3 章では順序回路をパルス駆動型ニューラルネットワーク上に実装する方式を述べる。第 4 章では、それを踏まえて実際に実装した順序回路について述べる。第 5 章では、今後の展開について述べ、最後に第 6 章でまとめを行う。

第 2 章　背景

本章では、人間のように知的なものを作る上で、従来の類似研究と比較して本研究のアプローチを述べる。その上で、本研究で神経網シミュレーションを行うにあたって採用したニューロンモデルについて述べる。

2.1　研究のアプローチ

人間の脳のように知的なシステムを作ることを考える。

そのようなものを目指す研究として代表的なものに人工知能研究がある。これは、人間の思考過程の内省に基づいた形での知識の単純な記号化を主とする方法であり、人間が明示的に与えるアルゴリズムによって人間の高位の思考過程を模倣できる。しかし、そのような方法では、人間がアルゴリズムを与えることができないことを実現することは不可能である。我々自身、自分の思考過程を考えると、脳は必ずしも論理的に説明可能な思考過程を経るとは限らない。よって、この人工知能研究の方法およびその延長線上に、脳のような知的なものを実現できるかどうかは疑問である。

一方、近年の大脳生理学の発展により、巨視的には機能局在の様子が、微視的には個々の神経細胞の働きがわかるようになってきた。PET(Positron Emission Tomography) や fMRI(functional brain Magnetic Resonance Imaging)[1] といった、無侵襲 (noninvasive) な高次脳機能計測の手法の発達により、肉体的な動作や思考を行っている時の脳内のおおまかな動作もわかってきている [2]。しかし、そのような手法が今後更に発達することによって、脳内の信号の動きなどが一つ一つわかったとしても、個々の神経細胞がどのように結び付くことによって脳の高次機能を実現しているのかは見えてこないのではないかと思われる。

このような状況から、我々は神経網のシミュレータを作成した。これは、上述のようなシステムを構築する際に、神経網の挙動を観測したり、脳のある機能を実現する際に考案するモデルの入出力が、実際の人間の振る舞いと同様であるかどうかを調べたりする際に必要となると考えられるからである。

そのため、シミュレートする脳の神経網の模倣品を構築する必要がある。しかし、我々は脳の神経網のようなものを作る方法を知らない。

そこで、まず調べたい脳の機能に関して具体的な問題を与え、その解法を論理的に構成してネットワークを作る。それを初期個体として、遺伝的アルゴリ

ズム (GA) によりその論理的構造を崩す。このようにして、シミュレートする神経網の模倣品を生成する。

我々は脳において記憶こそがその情報処理の本質であり、最も重要視すべき部分であると考える。よって、当面の目標は、まず人間の記憶のメカニズムを探るため、短期および長期の記憶がシミュレートされるような脳の神経網の模倣品を作ることとする。そして、シミュレーションによって得られた情報からその記憶における性質を解析し、「人間のように知的なシステム」を構成する際の参考にしたい [3]。

そのため、GA によって生成されるネットワークは記憶のための構造を持ったものであって欲しい。そこで、例として「任意の迷路を解くことができるネットワーク」を考える。ただし、俯瞰視点によってではなく、人間が大迷路の中を自分で歩いて解いていく時のように迷路を解くことを考えるものとする。問題としてこれを選択したのは、迷路を解くためには、自分がどちらから来たかというような短期記憶にあたるものと、迷路の解き方という長期記憶にあたるものの両方が必要になると考えるためである。

ここで迷路を解くアルゴリズムというのは十分具体的かつ論理的であり、それを実現するアルゴリズムは、論理式で表現することができる。それを順序回路として構成して、その論理素子をニューロンによって実現できれば、求めるニューラルネットワークを作成することができる。このように論理的に構成できるので、GA の初期個体として構成しやすいと考えられる。

このようなネットワークを作成し、それを初期個体として GA によって目的とするネットワークを生成する。このようにして生成されたネットワークならば、人間の脳に似たような振舞いをするだろうと考えられるため、目的とする記憶の構造を摑めるのではないかと考える。

2.2 ニューロンモデル

シミュレートする神経網の模倣品を構築するために、本シミュレータの採用するニューロンモデルは現実の脳の神経細胞に近いモデルでなければならない。そこで本節では、まず実際の神経細胞について述べ、その種々のモデルについて述べる。

2.2.1 実際の神経細胞

まず実際の神経細胞がどのようなものであるかについて述べておく。

神経細胞は、図1のように、核を含む細胞体から多くの突起が出ており、その中で最も長い一本が軸索であり、その他多数は樹状突起と呼ばれるものである。軸索は神経細胞の信号を送信する役割を持ち、樹状突起は信号を受信する役割を持つ。そして、樹状突起と他の神経細胞の軸索の末端との結合部をシナプスという。

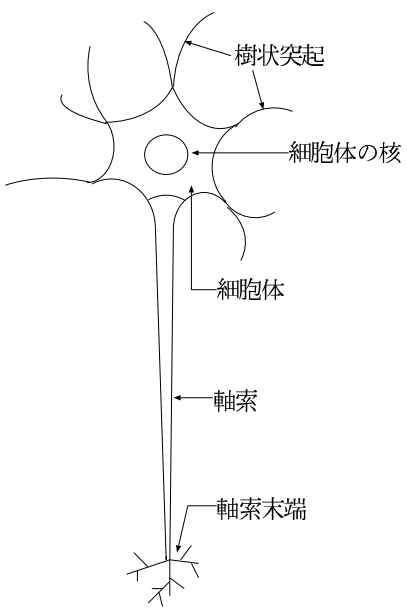

図1: 神経細胞

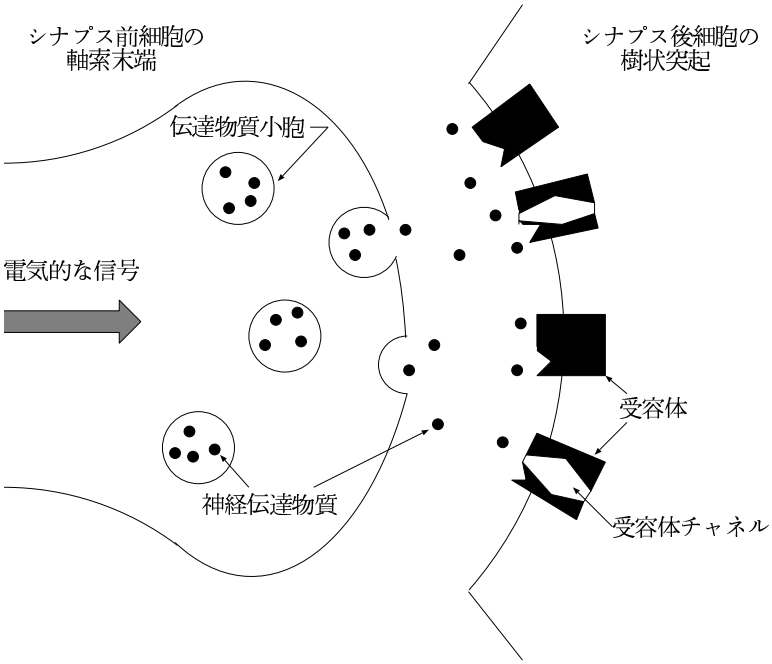

図2: シナプスにおける情報伝達

一般に、シナプスには隙間があり、電気信号のインパルスが軸索末端から樹状突起に直接伝わることはない。シナプスにおいて情報を伝える手段として使われているのが、神経伝達物質と呼ばれる化学物質である。以下、神経細胞間における信号の伝達方法について具体的に述べる。(図2)

神経細胞の軸索の膜にはナトリウムチャネルがあり、正電荷を持ったナトリウムイオン (Na^+) がここを通って流入することで、インパルスは発生する。

ところが、軸索の末端近くではナトリウムチャネルが減り、かわってカルシウムチャネルという別のタンパク質が増加する。そのため、インパルスがここに到達すると、軸索の末端には細胞の外部から大量のカルシウムイオン (Ca^{2+}) が流れ込む。

この結果として、伝達物質の詰まった伝達物質小胞が軸索末端で細胞の外に

開いて、伝達物質が放出される。シナプスの上流のシナプス前膜から放出された伝達物質は、下流のシナプス後膜の受容体と結合することでチャネルを開ける。

それにより、受容体を持つ神経細胞にイオンが流入して、シナプス後細胞の膜内外の電位差が変化する。この電位変化をシナプス後電位と呼ぶ。

シナプス後細胞の樹状突起には数百から数万個のシナプスが存在するので、それぞれのシナプス後電位は細胞体において総和される。

その総和された細胞体における膜電位がある閾値を越える程十分に大きければ、電位差を感じる性質を持つナトリウムチャネルにより、軸索にインパルスが発生する。

このような神経細胞をモデル化するにあたり、樹状突起の電導特性やイオンコンダクタンスなどの様々な特徴まで記述した複雑なモデルから、最も単純かつよく用いられるMcCulloch&Pittsによって提案された形式ニューロンモデルまで様々なものが提案されている。

実際の神経細胞に近い詳細なモデルは、神経細胞のかなりの性質を記述し得るものであるが、シミュレーションの際には計算量が膨大になりすぎる。また我々の研究自体が、単一の神経細胞に関する理解を目的とはしていない。知能は、中間的なスケールの物理現象であり、もう少し巨視的なレベルでの理解が必要であると考える[4]。

そこで、以下ではより抽象度を上げたモデルについて説明していく。

2.2.2 パルス駆動型ニューロンモデル

本シミュレータでは、実際の神経細胞よりも抽象度を上げたモデルとして、パルス到着の時刻や間隔の情報を扱う、実際のニューロンにより近いモデルであるパルス駆動型ニューロンモデルを採用している。以下、このモデルについて説明する。

時刻$t \geq t_0$におけるニューロンの膜電位を$P(t)$、閾値電位を$H(t)$とした場合、それぞれ以下のような式で与えられるとする。ここで、各変数/定数の添字E, Iはそれぞれ興奮性、抑制性を表している。

$$\begin{aligned} P(t) &= P_\infty + (P_0 - P_\infty)e^{-(t-t_0)/T_p} + \sum_{j=1}^{K_E} A_{E_j} \sum_{i=1}^{n_j} e^{-(t-t_{E_{ji}})/T_p} \\ &\quad - \sum_{j=1}^{K_I} A_{I_j} \sum_{i=1}^{m_j} e^{-(t-t_{I_{ji}})/T_p} \end{aligned}$$

$$
\begin{aligned}
H(t) &= H_\infty + (H_0 - H_\infty)e^{-(t-t_0-\tau_a-\tau_r)/T_h} \quad (t \geq t_0 + \tau_a + \tau_r) \\
&= H_0 + (H_{MAX} - H_0)e^{-(t-t_0-\tau_a)/T_r} \quad (t_0 + \tau_a \leq t \leq t_0 + \tau_a + \tau_r) \\
&= H_{MAX} \quad (t_0 \leq t \leq t_0 + \tau_a)
\end{aligned}
$$

P_0, H_0	初期リセット膜電位、初期閾値電位
P_∞, H_∞	静止膜電位、静止閾値電位
H_{MAX}	最大閾値電位
A_{E_j}, A_{I_j}	j番目のシナプスにおける単一パルスあたりの細胞膜電位の変化量
K_E, K_I	興奮、抑制それぞれの入力線路数
n_j, m_j	j番目の入力線路への入力パルス数
$t_{E_{ji}}, t_{I_{ji}}$	興奮、抑制パルスの印加時刻
t_0	前回パルス出力した時刻
τ_a	絶対不応期
τ_r	相対不応期
T_p	膜電位の時定数
T_r	相対不応期における閾値電位の時定数
T_h	相対不応期後の閾値電位の時定数

ニューロンの膜電位 $(P(t))$ はパルスが到着しない間、静止膜電位 (P_∞) に収束していく。パルスが到着すると当該入力枝に付加された重みの正負やその値に応じてその電位を急激に押し上げたり押し下げたりする。その後、また静止膜電位へと収束していき、それに達するとそこで膜電位は一定となる。

一方、ニューロンの閾値電位 $(H(t))$ は、単に静止閾値電位 (H_∞) に収束していき、達すればそこで一定となる。

パルスが到着した時に、膜電位の値が閾値電位の値を上回るとニューロンは発火する。この時、そのニューロンの膜電位は初期値 (P_0) にリセットされる。

リセットされたニューロンは不応期を経る。従来のパルス駆動型ニューロンモデルの場合は不応期を絶対不応期と相対不応期に分けることなく統一的に簡単化していたが [5]、本研究においてはより実際の脳に近づけるため、それらを分けて考えたモデルを用いている。

本研究におけるパルス駆動型ニューロンモデルは、リセットされたニューロンはまず絶対不応期を経る。この期間、ニューロンの膜電位はまたすぐに P_∞ に近づいていくが、閾値電位は最大閾値電位 (H_{MAX}) となる。この値は無限大

に近い値とする。そのため、どのような入力枝にパルスが入って膜電位がどれだけ上昇しても、ニューロンが発火することはない。

絶対不応期が終わったニューロンは次に相対不応期を経る。この期間、閾値電位が最大閾値電位から初期値 (H_0) へと収束していくので、閾値電位は通常の初期値から静止閾値電位へと収束していく場合よりも高くなっており、ニューロンは発火しにくくなっている。この相対不応期を経て、ニューロンは通常の活動を再開する。(図 3)

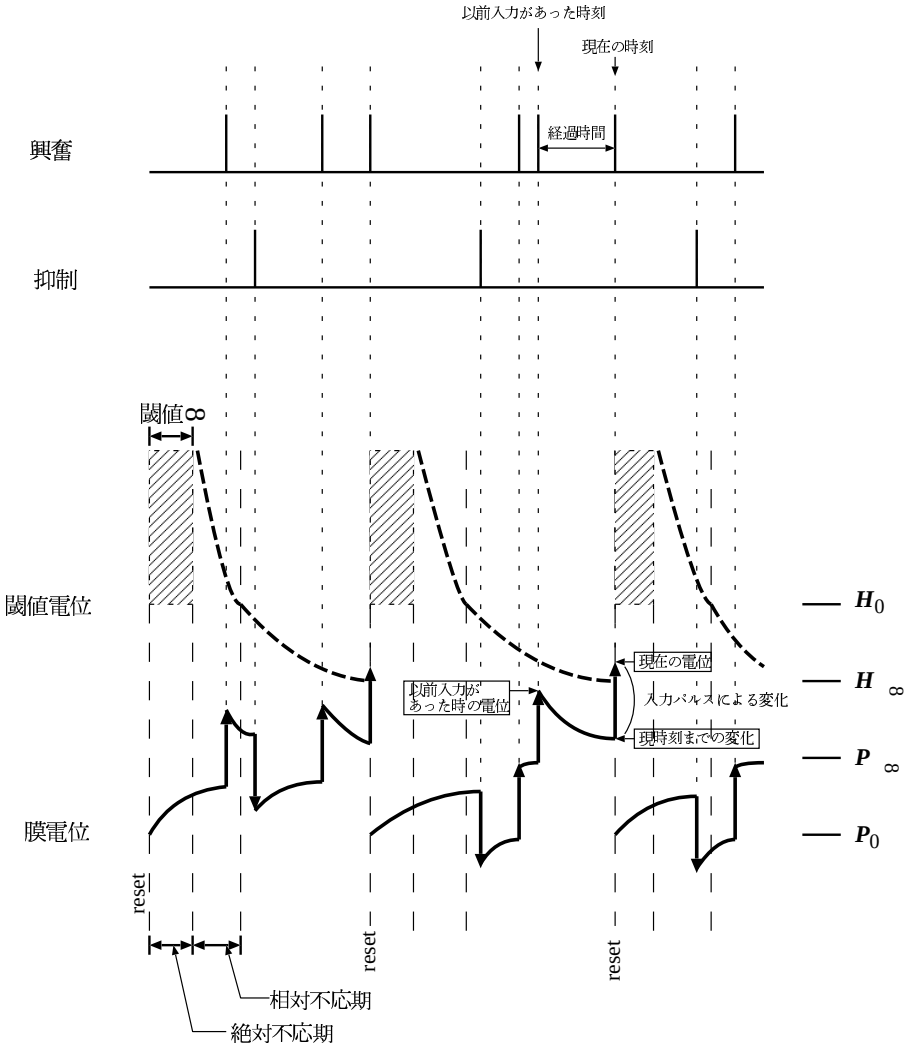

図 3: 入力の変化と膜電位の変化

このモデルでは入力がどのニューロンから来たかだけではなく、どの程度の時間間隔によって入って来たかという情報も用いて計算を行う。すなわち、入

力パルスの頻度が情報を表現している。

2.2.3 単純なニューロンモデル

パルス駆動型ニューロンモデルよりもさらに抽象度を上げた、最も単純でよく用いられるニューロンモデルが、McCulloch&Pitts によって提案された形式ニューロンモデルである。これは、ニューロンの平均発火頻度を入力とするモデル (図 4) である。

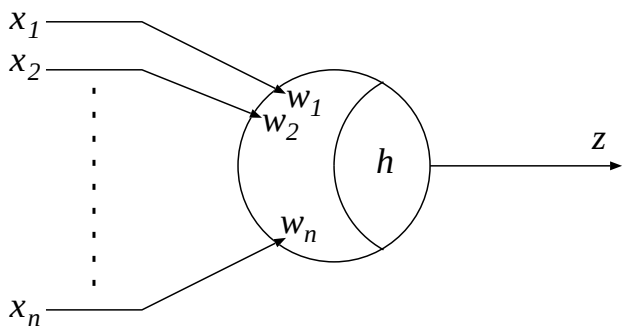

図 4: 単純なニューロンモデル

このモデルでは一つの神経細胞は n 個の神経細胞からそれぞれ $x_1, x_2, \cdots, x_n$ の強さの入力信号を受けとり、その膜電位 u および出力 zは

$$\begin{aligned} u &= \sum_{i=1}^{n} w_i x_i \\ z &= f[u-h] \end{aligned}$$

で表される。ここで h は閾値であり、$u-h \geq 0$ の時、そのニューロンは発火する。また、w_iはシナプスの結合効率を表す量で結合荷重と呼ばれるものであり、神経細胞の i 番目の軸索に単位入力が来たときに変化する膜電位の量を表す。興奮性細胞からのシナプスに対しては $w_i > 0$、抑制性細胞からのシナプスに対しては $w_i < 0$ の値をとる。

McCulloch&Pitts が提案した当初のモデルでは、神経細胞の評価関数 fには単位ステップ関数が用いられていた。この場合、入出力値は 0,1 のディジタル値をとる。

しかし、一般に fはシグモイド関数のような単調増加関数や単調非減少関数が用いられ、ニューロンの入出力は 0 から 1 のアナログ値をとり、そのニューロンの平均発火頻度を表現する。

このモデルでは、実際の脳の神経細胞における信号伝達において観測されているパルス情報などは考慮されていない。つまり細胞間を伝わるパルスの到着時刻や到着間隔という時間的な情報がこのモデルでは失われており、そのことがモデルの表現能力に限界を与えている可能性がある。

2.3 入出力特性による分類

以上のように様々なニューロンモデルがあるが、本研究において2.1で述べたような順序回路をニューロンで組むにあたり、論理素子をニューロンで実現する上で、論理値0/1に相当するようなパルス頻度は、簡単のため明確に区別できた方が良い。

実際、神経細胞はその入出力特性に閾値を持っており、出力が発火の有無という2値に別れているとみなすことができる。

そこで今度は、2.2で述べた種々のニューロンモデルの入出力特性に着目してみる。

具体的には、入出力特性がディジタルかアナログかによって分類する。

まず、単純な形式ニューロンモデルの場合であるが、入出力特性が図5のようにディジタルなものは、この神経細胞の評価関数を単位ステップ関数にすることで実現できる。単位ステップ関数は、入力の荷重総和

$$u = \sum_{i=1}^{n} w_i x_i - \theta$$

が0未満か0以上かによって、その出力である$f(u)$は0/1のディジタル値をとる。

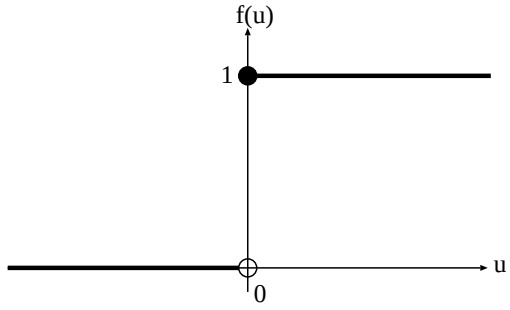

図5: 単位ステップ関数の入出力特性

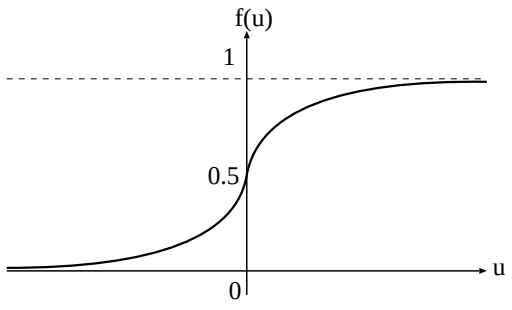

図6: シグモイド関数の入出力特性

また、入出力特性が図6のようにアナログなものは、神経細胞の評価関数$f(u)$

をよく用いられる先述のシグモイド関数とすることで実現できる。入力 u にアナログ値をとらせれば、出力もアナログとなる。

次に、本シミュレータで採用したパルス駆動型ニューロンモデルも、同様に入出力特性がディジタルかアナログかで分類できる。

入出力特性がディジタルなモデルとして図7のように閾値を持ったものがある。

このように閾値を持った入出力特性であれば、入力情報が最低頻度から最高頻度まで変化しても、出力情報は 2 値的に変化するために、論理値の 0 と 1 を明確に区別することができる。

また、実際の神経細胞の入出力特性も閾値を持ったものであるので、より実際の脳に近いシミュレーションを行うためにはこのようなモデルを採用する必要があるが、その実現は複雑である。

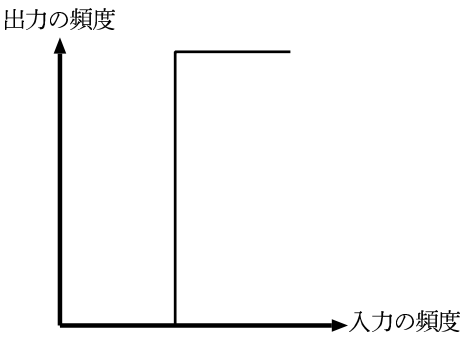

図 7: ニューロンの入出力特性 (2 値的)

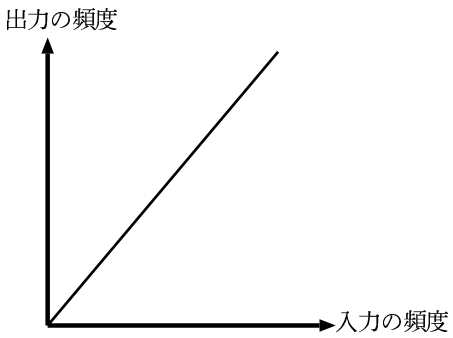

図 8: ニューロンの入出力特性 (線形)

そこで、実現が容易なものとして、入出力特性がアナログである、図 8のように線形なモデルがある。

このモデルであれば、入力パルスが入る度に発火するか否かを判断すれば、入力頻度の上昇に伴い出力頻度もそれに比例して上昇するため容易に実現できる。

しかし、このように線形な入出力特性では、入力頻度が最低頻度から最高頻度まで変化していくと、出力頻度がそれに伴って、入力頻度と同様な変化をするために、論理値の 0 と 1 が明確でないということが起こる。このため、論理値の 0 と 1 を明確にしたければ、入力頻度の与え方として、論理値 0 を最低頻度、論理値 1 を最高頻度というように 2 値的に与える必要がある。

そして、このような入力の与え方では、最低頻度と最高頻度の間の任意の頻度を入力情報とすることができない。つまり、最低頻度から最高頻度までの任

意の頻度情報を入力として与えるためには、図 7のように、入出力特性が閾値を持ったモデルでなければならない。

しかし、前述の通り、入出力特性がデジタルなモデルは実現は複雑であるため、本研究では入出力特性が線形のモデルを採用することにする。そのため、本研究で実現する順序回路に与える入力情報は 2 値的に与えることにする。

第 3 章　実装の方式

本研究は、第 1 章で述べたように、GA の初期個体を作成することが目的である。

よって本章では、そのための具体的な手順について先に述べ、次にそれをパルス駆動型のニューラルネットワーク上に順序回路として実装するために、ニューロンによって論理素子を実現する方法について述べる。

3.1　実装の手順

2.1節では、GA によって生成するネットワークの具体例として「任意の迷路を解くことができるネットワーク」を考えるとした。

しかし、GA の初期個体を作成する時に実際に考えるのは「簡単な迷路ならば解くことができるネットワーク」であるとする。これを用いて GA によって任意の迷路を解けるようなネットワークを生成する。

解かせる「簡単な迷路」とは、

- 東西南北の 4 つの方角のみに道がある。
- ゴールのある方角がわかっている。
- ゴールまでの道筋が唯一つ存在する。
- 単純にゴールの方角に向かっても、行き止まり等により辿り着けない。

というような性質を持つものであるとする。

ここで、ゴールの方角がわかっているようにするのは、ゴールがわからないのに進んでいくうちに偶然ゴールに辿り着いた、という無目的的な行動をとることを防ぎ、実際に人間が迷路を解く場合にゴールのある方向を認識していることがあるように、ゴールという明確な目的を持って迷路を解けるようにするためである。

本研究では、このような「簡単な迷路ならば解けるネットワーク」を作成す

ることを目的とする。そこで、まず「迷路を解く」という問題に対してその解法となるアルゴリズムを考え、そこから問題を解くための論理順序回路を求める。次にそれに基づいてGAの初期個体として、「簡単な迷路ならば解けるネットワーク」をパルス駆動型ニューラルネットワーク上に実装する。さらに、そこからより実際の脳に近いシミュレーションをするためのGAの初期個体を作るにはどうすればよいかについて考える。

3.2 ニューロンによる論理素子の実現方法

前節のような手順から、「簡単な迷路ならば解くことができるネットワーク」をパルス駆動型ニューラルネットワーク上に順序回路として実装するために、基本論理演算であるAND・OR・NOTをニューロンで実現するというアプローチをとった。

本研究では、シミュレーションのために時間を離散化している。パルス頻度を表す際、ある一定期間に全ての単位時間にパルスがあれば最高頻度とし、まったくパルスがなければ最低頻度としており、通常は頻度はこの間の任意の整数値を取り得るものとする。(図9)

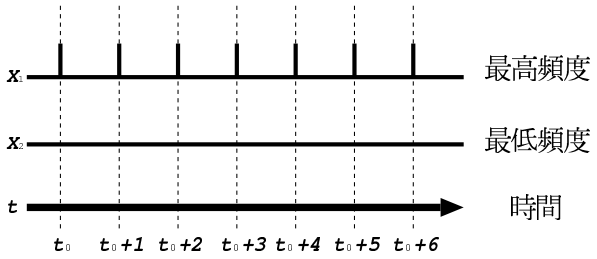

図9: 最高頻度・最低頻度

しかし、2.3節で述べたように、本研究では論理値0に相当する最低頻度、1に相当する最高頻度のみを入力パルスの列が表現する情報として与えることにする。よって、論理素子を実現するニューロンのモデルとして、簡単のため、その入出力特性が線形であるものを採用する。

このようなモデルとしては、図10のように膜電位と閾値電位の初期値と収束値を等しくし、その収束時間を1単位時間、不応期を絶対不応期・相対不応期共に0単位時間とすることによって、各単位時間に発火するかどうかが決定されるものが考えられる。そうすることにより、入力頻度が上昇するに従って、出

力頻度も上昇するという線形な特性を簡単に実現することができる。

よって、ニューロンが AND・OR・NOT のような働きをするモデルは図 10 のようになる。(図の黒丸は、そこで発火が起こることを示す。)

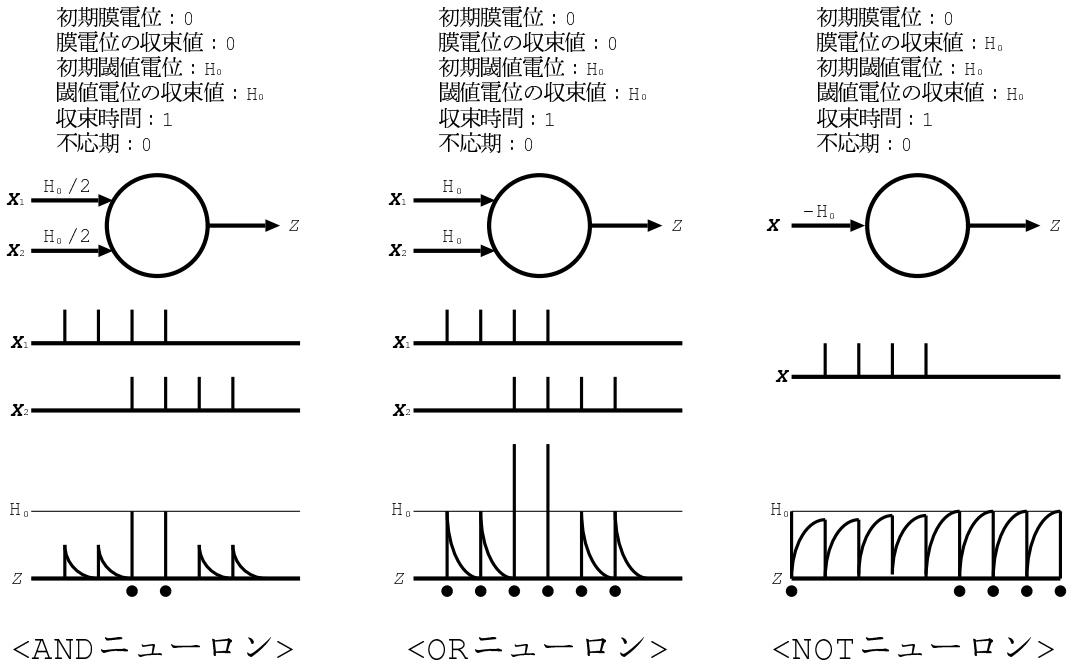

図 10: AND・OR・NOT ニューロン

ニューロンに AND 素子と同じ働きをさせるために、すべての入力枝に同時にパルスが入った時のみ発火するようにする。そのために、初期膜電位及び膜電位の収束値を 0 とし、初期閾値電位及び閾値電位の収束値を H_0 として、全ての入力枝の重みを「H_0/入力数」で与える。

例えば 2 入力の AND であれば、2 つの入力枝の重みとも $H_0/2$ となるようにする。このようにすることで、図 10 の左のように 2 つの入力 x_1, x_2 があって、x_1 か x_2 のどちらか片方のみに入力パルスがあってもニューロンの膜電位は閾値電位の半分までしか押し上げられないので発火しない。また、x_1 と x_2 の両方同時に入力パルスが入ると、膜電位はそれぞれの入力パルスによって閾値電位の半分ずつ押し上げられることになり、結果として閾値電位に達するので発火する。

ニューロンに OR 素子と同じ働きをさせるために、入力枝に 1 つでもパルスが入った時に発火するようにする。そのために、初期膜電位及び膜電位の収束値を 0 とし、初期閾値電位及び閾値電位の収束値を H_0 として、全ての入力枝の

重みを「H_0」で与える。

このようにすることで、図10の中央のように2つの入力x_1, x_2があって、x_1かx_2のどちらか片方のみに入力パルスがあるだけでもニューロンの膜電位は閾値電位まで押し上げられるので発火する。また、x_1と同時にx_2に入力パルスが入れば、膜電位はそれぞれの入力によって閾値電位分押し上げられるので、閾値電位の2倍の値となり、発火する。

ニューロンにNOT素子と同じ働きをさせるために、入力枝にパルスが入らない時のみ発火するようにする。そのために、初期膜電位を0、膜電位の収束値をH_0とし、初期閾値電位及び閾値電位の収束値をH_0として、全ての入力枝の重みを「$-H_0$」で与える。そうすれば、膜電位の収束値が閾値電位に等しいので、入力パルスによって膜電位が押し下げられなければ必ず発火することになり、論理素子のNOTと同様の働きを得ることができる。

このようにすることで、図10の右のように入力xがあって、入力パルスがないときはニューロンの膜電位は閾値電位に収束するため必ず発火し、入力パルスがあるときは膜電位を閾値電位分押し下げるので発火しない。

以上のように、各単位時間に発火するかどうかが決定されるため、図10のように、入力パルスが最高頻度か最低頻度のどちらかのみで入ってくれば、連続して発火するか全く発火しないかのどちらかである。よって入出力におけるパルスの頻度は、最低頻度か最高頻度のどちらかをとることになる。すなわち、頻度の表現する情報として、論理値の0/1が明確に区別される。

第4章　実装

本章では「簡単な迷路ならば解くことができるネットワーク」の具体的な実装方法について述べる。そのために、まず迷路を解く際に与える入力について述べる。次に、迷路を解くアルゴリズムを提示する。最後に、それを実現するために実際に構成した回路について述べる。

4.1　外部入力

まず、迷路を解く際の入力について述べておく。本研究において迷路を解く際の入力は、自分の周囲の4つの方角(北、東、南、西)の壁があるかないかという情報と、ゴールが4つの方角のうち自分の現在の位置からどの方角にある

のかを示す情報の 2 つである。迷路は俯瞰視点で解くのではなく、内部を進んで解くのであるから、壁の情報は迷路を解く上で絶対に必要な情報であることは間違いないだろう。また、ゴールのある方角という情報を与える理由は、3.1 節で述べた通りである。

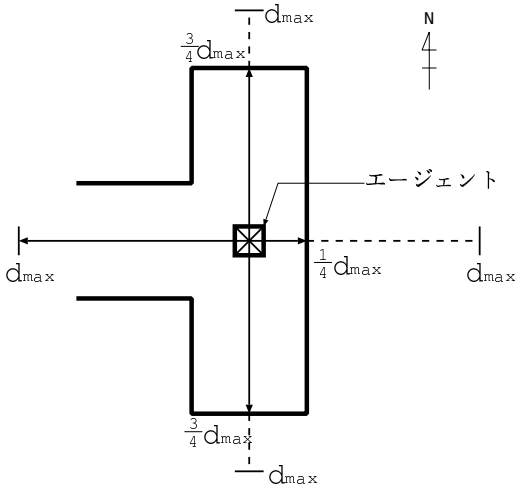

図 11: 壁の入力の例

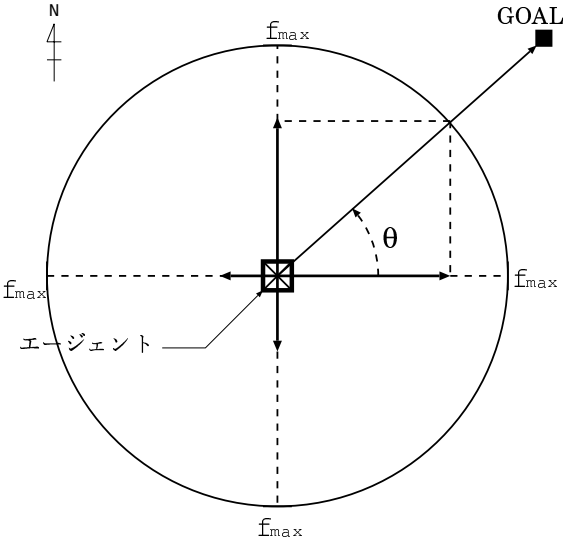

図 12: ゴールの方角の入力の例

壁の入力情報は 4 つの方角それぞれにおいて、壁を感知できる最大距離を d_{max}、エージェントと壁の距離を d とすると、壁の入力頻度は、ある定数 c があって $c(d_{max} - d)$ の値を頻度に変換して与えられる。すなわち、壁が近いほど入力頻度は大きくなる。ただし、$d_{max} - d < 0$ の時は、その値は 0 となる。つまり、各方角について、d_{max}以上離れた距離にある壁は認識できないものとする。

例えば、ある迷路の中にエージェントが図 11 のように存在するとする。この時、エージェントから壁までの距離は北と南が$\frac{3}{4}d_{max}$、東が$\frac{1}{4}d_{max}$、西は d_{max} よりも遠い。よってそれぞれの壁の入力情報は上式に従い、北と南は$\frac{1}{4}d_{max}$、東は$\frac{3}{4}d_{max}$、西は 0 をそれぞれ頻度に変換したものとなる。

ゴールの方角は、4 つの方角それぞれにおいて、その方角がゴールにどの程度近いのかを表す。例えば図 12 のような方向にゴールがあった場合、入力パルスの最高頻度の値を f_{max} とすると、

$$\begin{aligned} \text{北の入力頻度} &= f_{max} \sin\theta \\ \text{東の入力頻度} &= f_{max} \cos\theta \end{aligned}$$

$$\text{南の入力頻度} \quad = \quad f_{max} - \text{北の入力頻度}$$
$$\text{西の入力頻度} \quad = \quad f_{max} - \text{東の入力頻度}$$

となる。

すなわち、4 つの方角のうちゴールに近い方角については、θはゴールのに近い方角 (のうちの 1 つ) から実際のゴールの方向までの角度とすると、最大頻度f_{max}のうちの $\sin\theta$,$\cos\theta$となり、ゴールから遠い方角については、f_{max}からそれと反対の方角の入力頻度との差が与えられるとする。

今回与える入力情報は、3.2節で述べたように、最低頻度または最高頻度で与える。よって、それぞれの入力頻度は、上記のものをまるめて与えることにする。そこで、壁の入力情報は壁があれば最高頻度、なければ最低頻度、ゴールの方角の入力情報は、その方角にゴールが近ければ最高頻度、遠ければ最低頻度と考える。

4.2 迷路を解くアルゴリズム

迷路を解くための方法として最も簡単に思い付くであろう方法は、例えば左側の壁のみに沿って進むことである。確かにこの方法によってたいていの簡単な迷路は解くことができるだろう。しかし、「迷路を解くこと」が本研究の目的ではない。迷路を解く際に必要となる短期記憶や長期記憶の構造をつかむことが目的である。

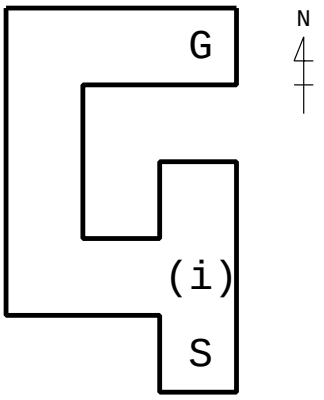

図 13: 簡単な迷路の例

では迷路を解く際、いったいどのような場面で記憶が必要となるだろうか。

例えば図 13のような迷路を上記 2 つの入力情報で解くことを考える時、まず

壁の情報から進む方角を選択するということが考えられる。スタート地点では壁の状況から北にしか行けないので北に進む。しかし (i) のような交差点では入力情報として上記 2 つしかなければ、よりゴールに近づくために北に進むだろう。しかし行き止まりなので再び (i) に戻ってくることになるが、今度は西と南のどちらに進めばゴールに近いのかわからない。

ここ、すなわち交差点で記憶が必要となる。(i) の交差点についてどちらの方角に自分が行ったことがあるかを記憶しておくことで、自分はすでに南にも北にも行ったことがあるので西に進む、といった選択が可能になる。

また、(i) の分岐から西に進んだ後、壁が北と南の 2 つの方角にしかないために、ここでも自分がどちらから来たのかということを記憶していないと同じところを行ったり来たりしてしまうことになる。

よって迷路を解く上で必要となる具体的な記憶は交差点の情報と、自分の後ろの方角という情報である。以上から、迷路を解くアルゴリズムは次のように決めることができる。

記憶を用いた「簡単な迷路を解くアルゴリズム」

以下の手順をゴールに辿り着くまで繰り返す。

1. 直進し、壁の状況が変化したときに「交差点」「曲がり角」「行き止まり」を判断
2. (a) 「交差点」の時は、
   - i. 初めて来た交差点なら、交差点の情報を記憶し、ゴールに近い方角に進む。
   - ii. 一度来たことがある交差点なら、その交差点の記憶と壁の有無からまだ行ったことのない方角を求め、その中からゴールに近い方角に進む。

   (b) 「曲がり角」なら道なりに進む。

   (c) 「行き止まり」なら引き返す。

4.3 回路構成

本節では、前節のアルゴリズムに従って構成した回路について、具体的に述べる。

4.1節で述べたような入力が与えられるものとして、4.2節で述べたアルゴリズムに従って、現在の入力から次に進む方角を決定する回路を構成した。図14に

全体のおおまかな構成を示し、以下それを説明する。

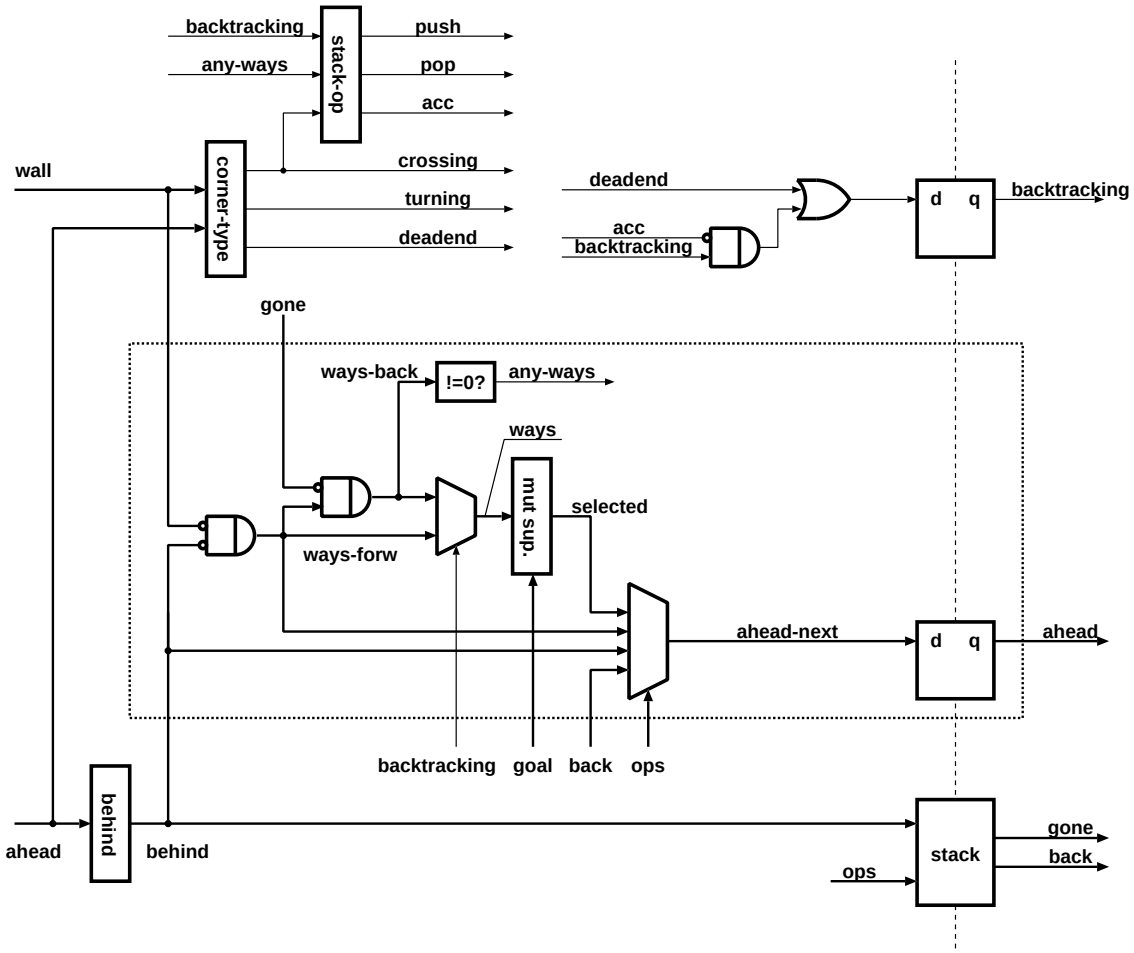

図 14: 全体図

入力は壁の情報である wall とゴールの方角である goal によって与えられる。出力は次の進行方向で、図 14では ahead である。この ahead は、出力であると同時に、現在の前を向いている方角を示す内部状態でもあり、計算にはこの状態も用いられる。周囲の壁の状況が変わらない限り次の進行方向の計算は行われないので、迷路内を動くエージェントは ahead の示す方角に進み続けるものとする。

4.2節のアルゴリズムの 1. に該当する計算を行う部分が、図 14のモジュール corner-type である。ここでは周囲の壁の状況が変化した時、すなわち直進路ではなくなり、ahead の示す方角に進み続けることができなくなった時に新たな計算

が行われる。その時の壁のある方角が 1 方向以下なら「交差点 (crossing)」、2 方向あってそのうち 1 つは前に壁があるなら「曲がり角 (turning)」、3 方向なら「行き止まり (deadend)」(図 15) と判断し、該当する計算の出力を制御するための信号を出力する。それと同時に、前述の ahead や後述する内部状態 backtracking や stack を変更するための制御信号として clk を出力する。それぞれの内部状態は clk の信号の立ち上がりによって更新されるものとする。

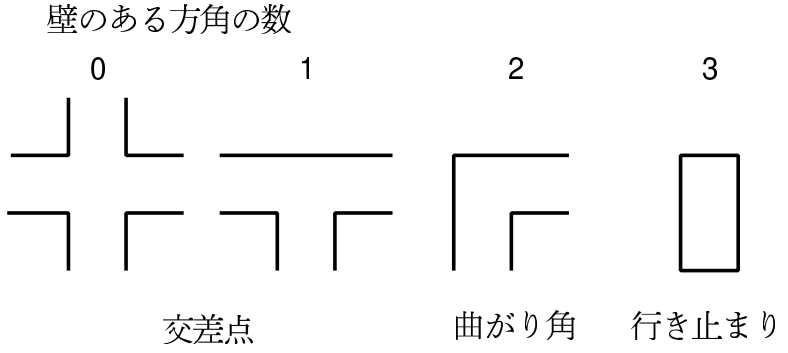

図 15: 壁の数による分類

ここで、「交差点」の時は、4.2節のアルゴリズムの 2.(a) から交差点の情報を記憶することが必要である。本研究ではこれをスタックを用いて行う。今回解く対象とする簡単な迷路では、通った経路のうち最近の交差点の情報しか用いないので、データ構造が LIFO であり、スタックで十分である。

またこの内部状態 stack の操作について、モジュール stack-op においてそのための制御信号が生成される。現在、行き止まり等から引き返しているかどうかを表す内部状態 backtracking と、その交差点からまだ行っていない道があるかどうかを表す信号 any-ways によって、stack の操作を制御するための制御信号 push,acc,pop は生成される。

初めてその交差点に来たならば、その交差点の情報をスタックに積んでやる必要があるため、

$$\mathsf{push} \quad = \quad \mathsf{crossing} \cdot \overline{\mathsf{backtracking}}$$

を出力する。

その交差点に戻ってきて、かつ、まだ他に行っていない道があるなら、現在

スタックトップの指すところに交差点の情報を追加してやる必要があるため、

$$\mathsf{acc} \quad = \quad \mathsf{crossing} \cdot \mathsf{backtracking} \cdot \mathsf{any\text{-}ways}$$

を出力する。

その交差点に戻ってきて、かつ、もう行っていない道はない時、その交差点の情報をスタックから降ろしてやる必要があるので、

$$\mathsf{pop} \quad = \quad \mathsf{crossing} \cdot \mathsf{backtracking} \cdot \overline{\mathsf{any\text{-}ways}}$$

を出力する。

ここで、push,acc,pop,turning,deadend の各信号は排他的に生成される。よって、回路全体の出力の制御は、これら 5 つの信号で行われる。図 14では、まとめて ops と表記してある。

内部状態 backtracking は、クロックの立ち上がりで動作するポジティブエッジトリガ形の D フリップフロップで保持されており、clk の立ち上がりでその値が更新される。「行き止まり」によって「引き返す (backtracking)」という状態になるので、deadend の信号によってセットされる。その後、交差点にまで戻った時に、そこから他のまだ行っていない道があれば、もう「引き返す」という状態ではなくなるので、acc の信号でリセットされる。ただし、交差点に戻った時に、もうその交差点から行ったことのない道がないという時は、先述した通り pop の信号が出力されるが、この時はさらに一つ前の交差点まで「引き返す」ため、backtracking の状態はリセットされない。

内部状態 stack の構造について詳しく述べる前に、「交差点の情報」とは具体的に何を指しているのかを明らかにしておく。ここで言う「交差点の情報」とは一度来たことのある交差点に戻ってきた時に、行ったことのある方角に何度も行かないですむようにするための情報、すなわち自分がその交差点から進んだことのある方角を指す。これを記憶しておくことで、同じ交差点で同じ方角に何度も進んでしまうことを回避でき、結果としてゴールに到達できる可能性が高まる。

この交差点の情報を蓄える stack は、その 1 段がスタックトップ (top) と、ある 1 つの交差点において自分が初めてその交差点にやって来た方角を記憶する部分 (back) とその交差点から行ったことがある方角を記憶する部分 (gone) の 2 つで構成されている。gone だけでなく back も必要となる理由は、その交差点

以降の経路のうちどの方角に行っても行き止まり等のためにその交差点に引き返してきた時に、gone の内容だけでは全ての方角に行ったことになっているだけで、最初にその交差点に入った方角だけを別に覚えていなければ、戻れないからである。gone、back 共に、それぞれ行ったことがある方角に 1、ない方角には 0 が積まれるとする。

また、スタックは 7 段とする。これはマジックナンバーが 7、すなわち、人間が瞬間的に把握して記憶できる対象の数は 7 個であると言われているところから、その段数とした。

スタックの更新操作は push と acc の 2 種類あって、更新データは後ろの方角を示す behind のみによって表される。push の時は、新しいスタックトップの back および gone に behind を書き込む。acc の時は、現在のスタックトップの gone に behind を OR する。図 16にスタックのデータ・アレイの全体図 (更新側のみ) を、図 17にそのセルの構成を示す。

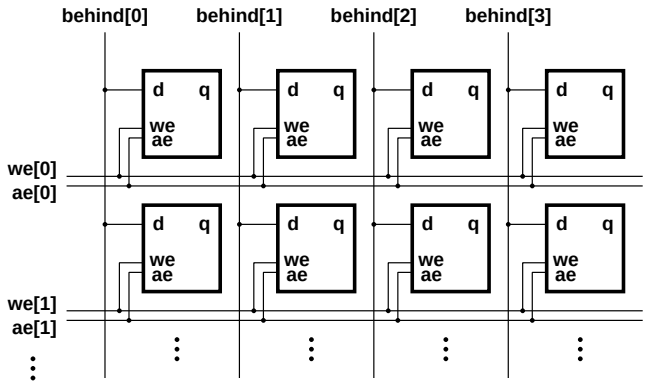

図 16: スタック・データ・アレイ (更新側のみ)

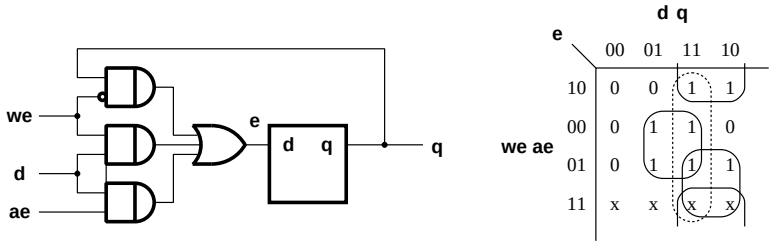

図 17: スタック・セル

各スタック・セルは、ポジティブエッジトリガ形の D フリップフロップを基本に構成されている。we は各段においてその一段下の top と push の AND である。また、ae は stack の back においては常に 0 であり、gone においては各段の top と acc の AND である。よって、push 時には we が、acc 時には ae がそれぞれ該当するセルにおいて 1 になって、その時にクロックが立ち上がれば、各セルの値が更新されるものとする。

以上のような入力、内部状態、制御信号から、出力である次の進行方向は以下のように計算される。

まず、ahead から、北と南、東と西を入れかえることによって、後ろの方角である behind を生成する。

もし corner-type で求まった制御信号が deadend ならば、4.2節のアルゴリズムの 2.(c) から、後ろに引き返すしかないので、この behind を ahead の次の更新データである ahead-next として選択するための候補とする。

また、$\overline{\text{wall}} \cdot \overline{\text{behind}}$によって turning の時の出力候補を生成できる。

これは同時に、初めてある交差点に入った時の出力の候補 (ways-forw) とすることができ、ある交差点に戻って来た場合には、$\overline{\text{gone}}$との AND をとることで、その場合の出力の候補 (ways-back) とすることができる。

そして、ways-forw と ways-back のどちらにより出力の候補を生成するかは、backtracking によって決定される。backtracking が 0 なら初めてその交差点に入ったものとして ways-forw を候補 (ways) とし、1 なら一度来たことのある交差点に戻って来たものとして ways-back を候補 (ways) とする。

この ways では、まだ行くことのできる道は一つとは限らない。よって、ゴールの方角を示す goal により、相互抑制を行う mut sup. によって、出力の候補となる方角を一つに絞る (selected)。

ここで、相互抑制とは、各方角について最も頻度の高いものを通すためのものであり、わずかでも頻度に差があれば、頻度の高いものを通して、頻度の低いものはそれによって抑制する。しかし、今回は入力情報が最高頻度と最低頻度で与えられるために、頻度に僅かの差がついたりはしないので、方角に優先順位を設けている。

以上により、制御信号が deadend なら behind、turning なら$\overline{\text{wall}} \cdot \overline{\text{behind}}$、push または acc なら selected、pop なら back を選択し、ahead-next として内部状態 ahead を保持するポジティブエッジトリガ形 D フリップフロップに入力する。

その後、clk が立ち上がることにより、ahead は更新される。

以上のように、次の進行方向が計算される。

第 5 章　提案

以上のように、入力が 2 値的に与えられるモデルで「簡単な迷路ならば解くことができるネットワーク」を順序回路として実装した。

しかし、実際の神経細胞の入出力特性が閾値を持っていることから、より実際の脳に近いシミュレーションを行うためには、そのような入出力特性を持った論理素子の振る舞いをするモデルを与えるべきである。

そして、入力情報はパルスの最低頻度と最高頻度の 2 値的にではなく、その間の任意のパルス頻度をとることができるようにするべきである。

そのためには、2.3節で述べたように、閾値を持った 2 値的な入出力特性を持つニューロン、すなわちある頻度を閾値として、それ以下の頻度であれば発火せず、それ以上ならば一定の頻度で発火するようなニューロンを作成すればよい。そのようなニューロンは、例えば次のように作ることができる。

ある頻度において図18のように、ぎりぎり発火せずに安定するような重みが入力枝に付加されているとする。つまり、入力は、入力パルスが一度入って膜電位を重みの分押し上げた後、その膜電位がぎりぎり次の入力パルスが来ても発火しないところまで下がってから次の入力パルスが入るような頻度であるとする。ここで、図18,19において、縦軸は膜電位、横軸は時間、中央の横線は膜電位の収束値、一番上の横線は閾値電位である。

この時、この入力頻度を僅かに高くすると、図19のように膜電位がぎりぎり下がりきらないうちに次の入力パルスが入っているため、発火するようになる。そして、発火後は初期膜電位にリセットされているため、次の入力パルスでは発火せず、さらに次の入力パルスで発火する。これらから、この頻度以下で入力パルスが来れば全て発火せず、この頻度以上で入力パルスが来るならば 2 回に 1 回発火することがわかる。

このニューロンの入出力特性を図20を用いて説明する。

この図において、入力頻度 x が 0 から上昇しても閾値 f_θ までは発火しないので出力頻度 y は上がらない。この区間を**遮断領域**と呼ぶことにする。

x が f_θ を越えると、2 回に 1 回発火するので入力頻度と出力頻度との関係は

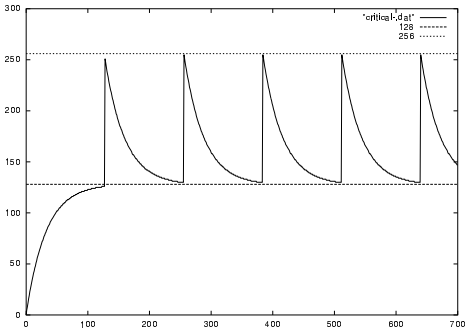

図 18: 遮断領域

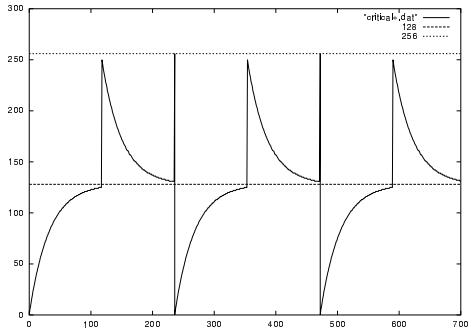

図 19: 線形領域

$y = \frac{x}{2}$の直線上にのる。この区間を**線形領域**と呼ぶことにする。

本ニューロンモデルにはある一定時間の不応期があるので、x がある頻度 f_{ref} 以上より上がっても、不応期の間は入力パルスがいくつ来ても発火に寄与できないため、出力頻度はそれ以上上がらないことになる。この区間を**飽和領域**と呼ぶことにする。

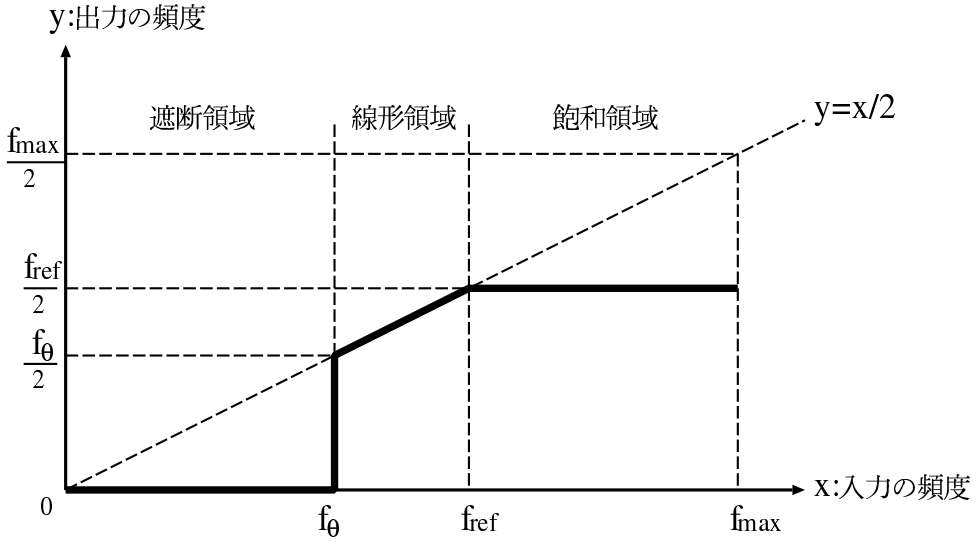

図 20: ニューロンの入出力特性

ここで注意すべきことがある。これはあくまでも入力頻度が閾値を越えた時の発火が 2 回に 1 回起こるように重みを設定した場合の例であるから、線形領域の特性が $y = \frac{x}{2}$ となるのであり、もし重みがそれより小さくすれば、当然発火までに必要な入力パルス数は増加し、それに伴いその特性の傾きは小さくなる。

例えば、図 21 のように先の場合よりも小さな重みでぎりぎり発火せずに安定

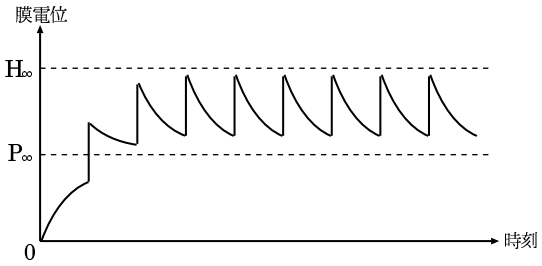

図 21: 遮断領域

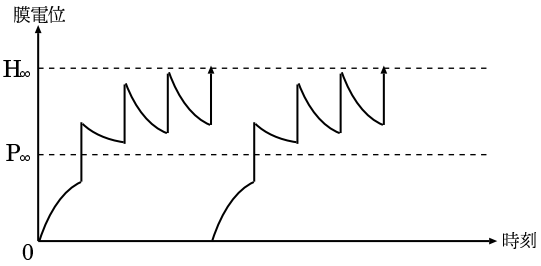

図 22: 線形領域

する場合、僅かに入力頻度を高くしてやると図22のように入力頻度が閾値を越えた時の発火が4回に1回起こる。この例では線形領域の特性は $y = \frac{x}{4}$ となる。

ただし、2回に1回以上の割合で発火することはないので、線形領域の特性の傾きは$\frac{1}{2}$以上になることは有り得ない。なぜなら、1回で発火するような重みは、膜電位の初期値と収束値の差よりも大きいことになるため、ある入力頻度によってぎりぎり発火せずに安定させることができるような閾値を持つことができないからである。

第4章で実装した回路について、論理素子の振舞いをするモデルの入出力特性が線形であるものから、このような閾値を持ったものに換えてやることによって、より実際の神経細胞に近いニューロンモデルによるネットワークにすることができる。

しかし、このように閾値を持つニューロンが最高でも2回に1回の割合でしか発火しないので、入力パルスの頻度は常に低下してしまうことになる。そこで、頻度を増強してやる機構も必要となってくるだろうと思われる。それも今後の課題である。

第 6 章　おわりに

本論では、人間のように知的なシステムを構築することを目標とする研究の準備として、脳で最も重要であろうと思われる記憶の構造を摑むため、神経網の模倣品を GA によって生成するための初期個体を順序回路としてパルス駆動型ニューラルネットワーク上に構成した。

まず、我々の研究の方向性を明確にするため、従来の類似の研究に対する疑問を提示し、我々は神経網シミュレータ上で、GA によって生成した神経網の模倣品を用いて研究することを述べた。

そこで、実際に我々が用いるニューロンのモデルであるパルス駆動型ニューロンモデルについて述べ、他のモデルではなくそのモデルを採用した理由を述べた。

さらに、そのモデルによって論理素子を実現する際に、まず入出力特性によって分類できる様々なモデルを示した。その中で今回は、簡単のために順序回路に与える入力を2値的にしたため、その実現の容易さから入出力特性が線形であるモデルを採用した。

そして「迷路を解く」という具体的な問題に対しアルゴリズムを与え、それを論理式で表現して、パルス駆動型ニューロンモデルで実現した論理素子を用いて実際にそれを順序回路としてネットワーク上に構成した。

最後に、その時に用いた論理素子の振舞いをするニューロンのモデルをより実際のニューロンに近づけるための方法を提案した。それを用いて論理素子の振舞いをするニューロンのモデルを作成し、今回作成した順序回路に適用すれば、より実際の脳に近いシミュレーションができると考えられる。

謝辞

本研究の機会を与えて下さり、適切な御指導を賜わりました富田眞治教授に深甚な謝意を表します。

また、貴重な御助言をいただいた森眞一郎助教授、五島正裕助手、津邑公暁氏、三輪忍氏に心から感謝いたします。

さらに、日頃暖かく御鞭撻下さった京都大学工学部情報学科富田研究室の諸兄に感謝致します。

ありがとうございました。

参考文献

[1] Ogawa, S., Tank, D.W., Monon, R., Ellermann, J.M., Kim, S.G., Merkle, H. and Ugurbil, K.: Intrinsic Signal Change Accompanying Sensory Stimulation: Functional Brain Mapping with Magnetic Resonance Imaging, Vol.89, USA, Proc. Natl. Acad. Sci., pp.5951-5955(1992)

[2] Tovee, M.J. and Cohen, E.M.:Working memory: Trouble in mind, Current Biology, Vol.6, No.13(1996)

[3] 津邑公暁:神経網の高速シミュレーション手法 (1998)

[4] 津邑公暁, 三輪忍, 五島正裕, 富田眞治:記憶構造観測のための神経網シミュレーション, 第 20 回 計測自動制御学会システム工学部会研究会『人工生命の新しい潮流』 pp.111-114(2000)

[5] Segundo, J.P.,Parkel, D,H., Wyman, H., Gegstad, H. and Moore, G.P.:Input-output relations in computer-simulated newve cells, Kybernetic, Vol. 4, No. 5, pp.157-171(1968)