GPGPUによる様々なニューロンモデルのシミュレーションの高速化と計算精度の評価

奥野 舜† 藤田 一寿†‡ 樫森 与志喜†

†電気通信大学情報理工学研究科基盤理工学専攻  
‡津山工業高専専門学校情報工学科

E-mail: †o1633021@edu.cc.uec.ac.jp, ‡k-z@nerve.pc.uec.ac.jp, †kashi@pc.uec.ac.jp

**あらまし** 脳における感覚情報の処理機構を理解するためには実際のニューロンを模した複雑なニューロンモデルから成る大規模なネットワークモデルをシミュレーションする必要がある。しかしながら、このようなシミュレーションは極めて長い計算時間を必要とし、現実的な時間スケールでのコンピュータシミュレーションを行うことはできない。この計算時間の問題を解決するために、我々はGPGPUによる計算時間の短縮に注目し、それはきわめて多くのニューロン数のシミュレーションに有効な方法を与える。本研究では、我々はニューロンモデルに対して計算を並列に実行するGPGPUの計算法を開発した。この方法を用いて、GPGPU計算法がニューロンのシミュレーションの計算時間を優位に減少することを示した。また、単精度と倍精度の計算はほとんど結果に差を生じないことも示した。これらの結果は、単精度でのGPGPU計算が大規模なニューラルネットワークのシミュレーションに対する最も有効な方法であることを示唆する。

**キーワード** GPGPU, 高速化, 計算精度, 神経シミュレーション

GPGPU-accelerated simulation and accuracy evaluation for various neuron models

Shun OKUNO† Kazuhisa FUJITA‡ and Yoshiki KASHIMORI†

†University of Electro-Communications   
‡National Institute of Technology, Tsuyama College

E-mail: †o1633021@edu.cc.uec.ac.jp, ‡k-z@nerve.pc.uec.ac.jp, †kashi@pc.uec.ac.jp

**Abstract**　To understand the processing mechanisms of sensory information in the brain, it is necessary to simulate a huge size of network that consists of complicated neuronal model imitating the actual neurons. However, such a simulation requires a very long computation time, failing to perform computer simulation with a realistic time scale. In order to solve the problem of computation time, we focus on the reduction of computation time by GPGPU, providing an efficient method for simulation of huge number of neurons. In this paper, we develop a computational architecture of GPGPU, by which computation of neurons is performed in parallel. Using this architecture, we show that the GPGPU method significantly reduces the computation time of neural simulation. We also show that the simulations with single and double float precision give little significant difference in the results, independently of the neuron models used. These results suggest that the GPGPU calculation with single float precision is most efficient method for simulation of a huge size of neural network.

**Keywords** GPGPU，Acceleration，Accuracy evaluation, Neural simulation

1. イントロダクション

生物の感覚系の実験では刺激や神経細胞活動の正確な計測が難しく、コンピュータシミュレーションがそれらの知見を得る有効な手段となる。しかしながら、コンピュータシミュレーションを行う上で現実に即した刺激に対する神経細胞の応答解析を生物の脳を模した大規模なニューラルネットワークにおいて行う場合、計算機に膨大な計算時間がかかってしまい、現実的ではない。このような理由から、現実的な計算時間を要する小規模で簡易な系のみを研究対象としていた。

しかしながら、近年コンピュータ技術の発展により、三次元画像処理専用ユニットGPU(Graphics processing unit)の性能を数値計算に対して汎用的に適用するGPGPU(General Purpose computing of GPU)が一般的に普及してきた。GPUの特徴として、シンプルな演算ユニットを大量に持っており、これを用いて効率的に並列計算を行うことができる。大規模なニューロン計算においてはCPUシングルコアによる計算に対して、非常に高いパフォーマンスを得ることができる。現在特に普及しているGPGPUの環境は、nVidia社の提供するCUDAとApple社により提案されたOpenCLに二分される。本研究では、nVidia社が開発したグラフィックカードのうち、比較的安価であるモデルのGeforce GTX 750 Tiを用いてCUDAによる計算時間の特徴を調査した。さらに、OpenCLの一種であるOpenMPも計算時間短縮の有力な手段である。OpenMPはマルチコアCPUを対象とした技術であることから、CUDAと比較して並列化可能なスレッド数は低くなってしまう。しかし、CUDAではGPUアーキテクチャの特徴を理解した上で最適なパフォーマンスを発揮できるようなプログラムを作成する必要があったが、OpenMPは既存のプログラムにコマンドを書き入れるだけで並列化が可能となる。その導入の簡便さから、情報技術に不慣れな研究者にとっても安易に利用できる点で非常に優れている並列化手法である。

GPGPUを使ったネットワークの計算時間の研究に関しては、Pallipuramらが簡単な1次元2層ネットワークを用いて評価している[1]。今回は、より現実的な感覚系のネットワークの例として、電気魚の電気感覚系[2]のモデルを使用した。これにより、実際の感覚系に即した大規模なモデルを想定し、それに最適化された並列計算により計算の効率化を図る。それによる高速化をCUDA、OpenMPおよびシングルコア計算を対象としてそれぞれ比較を行った。さらに、Leaky integrated-and-fire model, Izhikevich neuron model, Hogkin-Huxley neuron model, 2-compartment neuron modelの4種のニューロンモデルを対象とし、それぞれの計算時間を比較した。また、浮動小数点演算における単精度・倍精度の場合の計算時間の差異についても調べる。

1. 電気定位

今回の研究において、弱電気魚の電気定位をモデルとして神経細胞の活動を取り扱う。弱電気魚は自己発電器官を自身の尾部に所持しており、その電気出力EOD(Electric organ discharge)によって自身の周囲に電場を生成する。このEODと体の周辺に存在する物体によって発生するEODの変調EOD AM(EOD amplitude modulation)を体表面上の電気受容器で検出し、それを後脳の電気感覚側線葉と中脳の半円隆起に存在する神経細胞によって処理し、自身の周囲の情報を得ることが知られている[3]。

1. 手法
   1. 刺激入力のモデル

2節で述べたように、弱電気魚は自身の自己発電器官により体表面を中心として周囲に電場を形成する。これによる電気刺激は、電気受容器細胞への入力となり、電気受容器細胞は電気刺激をスパイクへと変換して上位の神経細胞へと送る。今回の研究では、電気受容器は受けたEOD AMに応じた入力を上位の神経細胞へと送るというモデルとなっている。

今回、電気受容器は弱電気魚の体表面を頭から尾の方に向かって一次元配列を取っている。

電気受容器が受ける電気刺激の電位*v*は式(1)のように表される。

(1)

*c*は皮膚の抵抗値*、i*は皮膚を通過する電流を表す。これにより受けた電気刺激のうちEOD AMに応じたスパイクを上位の神経細胞へと伝達する。ここで、*i*番目の神経細胞へと伝搬する入力*I*は式(2)のように表される。この電流が正弦波で表現されることは、弱電気魚が交流電場を生成することに起因する。

(2)

また、この入力は時間的に変動する。電気魚の頭から尾に向かって物体が移動し、EOD AMが変化していく様子を正弦波とガウス関数にてモデル化している。

* 1. 神経細胞のモデル

本研究では、電気受容器の上位核として位置づけられる様々な神経細胞モデルに対して刺激の入力を行い、それぞれのモデルに応じた計算時間の特徴の調査を行う。今回採用したニューロンモデルはLeaky integrated- and-fire model, Izhikevich neuron model, Hogkin-Huxley neuron model, 2-compartment neuron modelの4種であり、それぞれのモデルの概略を以下に示す。また本研究ではN個の神経細胞を持つ神経ネットワークについて、1000ms間のスパイク活動を計算する。ただし、ここで神経細胞間の互いの接続は存在しないものとした。

* + 1. Leaky integrated-and-fire model

　Leaky integrated-and-fire (LIF) modelは数ある神経細胞モデルのうち最も簡単なものの一つであり、神経細胞の膜電位を１階の常微分方程式で表現したモデルである。

(3)

　τは時定数、*I(t)*は入力電流を表す。膜電位*V(t)*が細胞の発火閾値θを超えたときに神経細胞は発火し、膜電位*V(t)*はリセット電位へ戻る。入力電流である*RI(t)*は、式(2)に即した値が与えられ、この時の*R*=15mVとしている。

* + 1. Izhikevich neuron model

Izhikevich neuron modelは2003年にE.M. Izhikevichにより提唱されたモデル[4]であり、式(4)および式(5)に示す2階の常微分方程式で表される。比較的単純なモデルながら、複数のパラメタによって様々なニューロンの応答を再現できるという利点を持つモデルである。LIFモデルと比較したとき、格段に実際の神経細胞に近い挙動を示すことが示されている。

(4)

(5)

ここで、*I*は入力、*v*は膜電位、*u*は回復変数を表す。細胞が発火閾値*θ*を超えた際、膜電位はパラメタcへリセットされる。Izhikevich neuron model最大の特徴として、細胞ごとに内因する膜特性を再現するために回復変数というパラメタが導入されている。この回復変数について、細胞が発火したとき一時的に発生する膜電位を安定化させる力が働くとして、*u=u+d*のように値を加算する。これに用いられている*a,b,c,d*の4つのパラメタを調整することにより、K+チャネルの活性化・Na+チャネルの不活性化などの細胞内因性の膜特性を表現することが可能となる。今回用いたパラメタとしては、a=0.02,b=0.2,c=-65mV,d=8,*θ=*20mVである。これは最も一般的なスパイク応答を示すRegular Spikeを生じるパラメタ値である。

* + 1. Hodgkin-Huxley model

Hodgkin Huxley modelは、膜電位固定法という実験手法の結果に基づき、HodgkinとHuxley[5]によって考案された、神経細胞の活動電位生成メカニズムを可能な限り再現した神経細胞モデルである。これは4つの常微分方程式にて表され、膜電位VとNaチャネルの活性化変数h、Naチャネルの不活性化変数n、およびKチャネルの活性化変数nの相互作用によって膜電位が定まる。これは式(6)から式(9)のように表される。

(6)

(7)

(8)

(9)

また、式(7)から式(9)にて用いられる*α、β*は膜電位依存性の関数であり、式(10)より式(12)にて表される。

(10)

(11)

(12)

このうち、は入力電流を表し、式(2)にて表される。それぞれのパラメタの値は、K.Wada と Y.Sakaguchiによるニューロンの連続発火メカニズムについての論文 [6]に記載されている値を採用し、膜容量=1μF/cm2、それぞれの最大コンダクタンスは=120 mS/cm2, =55mV、=36 mS/cm2, =-77mV、=0.3 mS/cm2, =-54.387mVとした。

* + 1. 2-compartment model

2-compartment modelは、2002年にB. Doironらによって考案された[7]モデルである。Hodgkin-Huxley modelを基にして、特に弱電気魚の神経細胞に見られる、錐体ニューロンと呼ばれる1つの神経細胞をSoma部位とDendrite部位の2部位に分けてそれぞれの膜電位が相互干渉し発火する。Soma、Dendriteそれぞれの膜電位に加え、それぞれのNaチャネルの活性化変数,

、不活性化変数、Soma側におけるKチャネル不活性化変数、Dendrite側におけるKチャネル不活性化変数の、時間遅延を持ったKチャネルの不活性化変数の要素を持ち、今回の研究で用いるニューロンモデルの中では最も複雑な系となっており、6つの常微分方程式にて表され、式(13)から式(18)にて表される。

(13)

(14)

(15)

(16)

(17)

(18)

またこれらの式中で用いられる関数は式(19)から(24)にて与えられる。

(19)

(20)

(21)

(22)

(23)

(24)

このうち、はSomaへの入力電流を表し、式(2)にて表される。それぞれのパラメタの値は、B.Doironらの論文[7]に記載されているものを採用した。

* 1. 処理の流れ

CPUにてシングルコア(並列計算を行わない)場合とOpenMPを用いた場合による処理の流れの比較をFig. 1に示す。iは各ニューロンの番号を表し、Calculate Potentialは各ニューロンモデルでの膜電位計算プロセスを示す。OpenMPでは、この膜電位計算プロセスについて並列化し、計算時間を計測する。

Untitled Diagram (1)

Fig. 1 フローチャート図(CPU)

次に、GPUの場合の処理の流れをFig. 2に示す。GPU演算の場合では、各ニューロンの膜電位計算をGPUによって並列処理する。この際、GPU側で演算を行った結果を返り値としてCPU側に返さなければデータの出力を得ることができないが、このCPU-GPU間でのデータ転送がボトルネックとなり計算時間の高速化に悪影響を及ぼしていることが知られている[8]。

GPU

Fig. 2 フローチャート図(GPU)

* 1. 実験環境

実験に用いた計算機のスペックを以下に示す。

CPU: Intel® Xeon E5-2630 v2, 2.60GHz

CPUメモリ: DDR3 16GB

GPU: nVidia Geforce GTX750Ti

CUDA コア数: 640個

プロセッサ周波数: 1020MHz

GPUメモリ: GDDR5 2GB

GPUメモリ帯域: 86.4GB/sec

本研究では、すべてのプログラムはLinuxディストーションの一つであるUbuntu 16.04上で開発し実行を行い、CUDA開発環境はバージョン7.5のものを使用し、GPU用プログラムを開発した。また、CPU演算用のプログラムはC++にて開発しGCC/G++ 5.4.0を用いてコンパイルされた。

1. 結果
   1. 各ニューロンモデルの計算時間比較

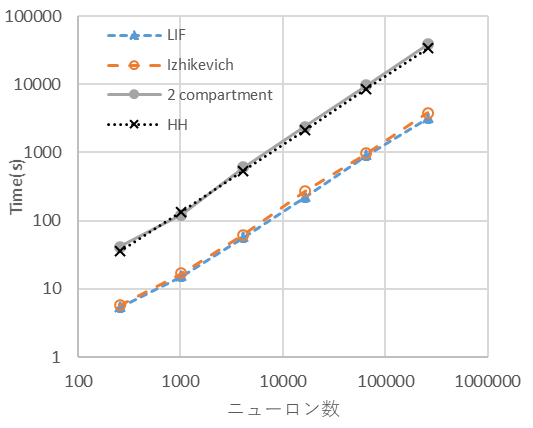
電気魚における神経細胞の活動を計算する部分にて、各ニューロンモデルについてニューロン数を2n (n= 8, 10, 12, 14, 16)と変化させながらシミュレーションを行った。Fig. 3-5ではシングルコア, OpenMP, CUDAのそれぞれの場合の実行時間を場合を示す。各図のそれぞれの場合でLIF, IzhikevichとH-H, 2-compartment modelの実行時間が約10倍異なるという特徴が見られた。また、CUDAの実行時間が他より短くなっている。今回の計算機において、CPUが比較的高価なXeon E5-2630 v2, 2.60GHz ×12であることとGPUが非常に安価なGeforce GTX 750Tiであることを考慮すると、CUDAによる高速化は非常にコストパフォマンスに優れた並列化手法であることが言える。但し、ニューロン数が1000個を下回るような小規模計算の場合においては、OpenMPの方が高速である。

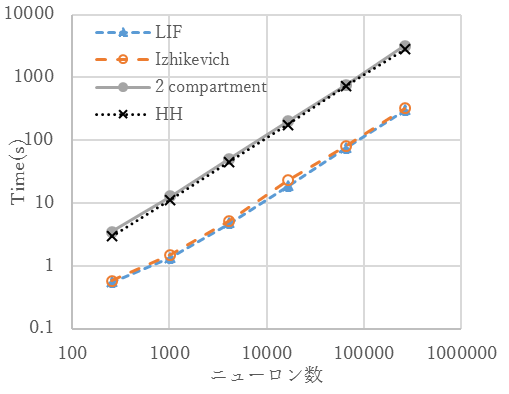
* 1. データ転送による計算時間比較

次に、各ニューロンモデルのCUDA計算において、CPU-GPU間のデータ転送を排除した場合の計算時間比較をFig. 6に示す。Fig. 4およびFig. 5の結果では、ニューロン数が少数の場合においてOpenMPの方が高速であったが、データ転送を排除した場合ではニューロン数が少数の場合においてもCUDAの方が高速であるという結果が示された。

* 1. 単精度、倍精度の場合の計算結果比較

CUDA並列計算にて、計算量の差が最も大きい2- compartment modelとLIF modelを対象に、単精度floatの場合(A)および倍精度doubleの場合(B)を用いた場合の計算時間の比較をFig. 7に示す。その際の2- compartment model のあるニューロンの0msから100msまでの膜電位変化をFig. 8に示す。また、Fig. 9 には2- compartment modelの0msから1000msまでのスパイク発火のISI(Inter spike interval)を、単精度(A)、倍精度(B)の場合に示す。これらに示される結果からは、単精度の場合と倍精度の場合で膜電位変化、ISIには有意な差が生じていない。LIFモデルについても、同様に単精度と倍精度に大きな差は存在しなかった。これより、神経モデル計算においては、単精度の方が倍精度と比較し効率のよい計算が可能である。

****

Fig. 3 非並列時の実行時間比較

**Fig. 4 OpenMPによる実行時間比較**

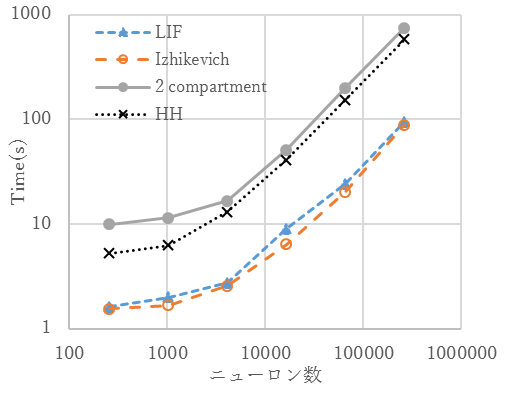
****

Fig. 5 CUDAによる実行時間比較

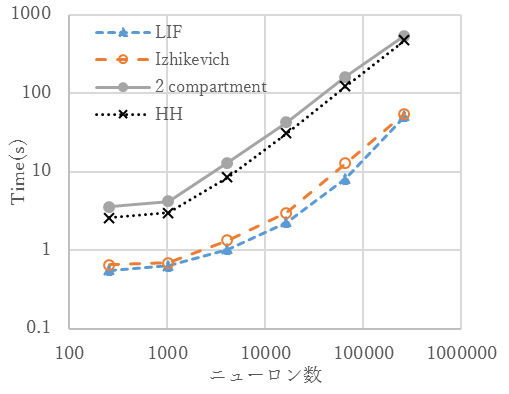


Fig. 6 データ転送による計算時間への影響

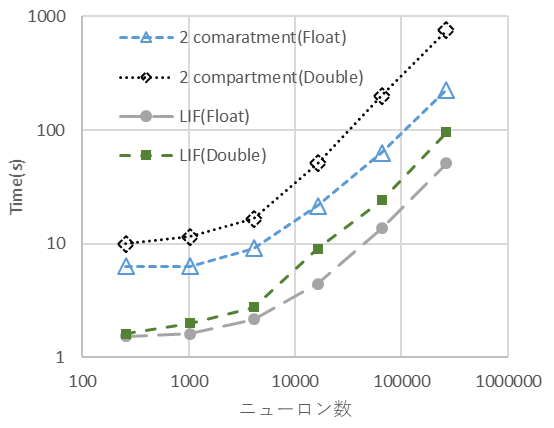
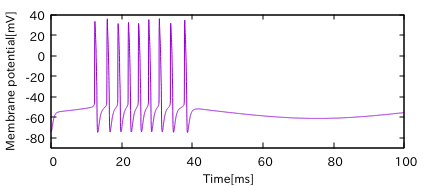
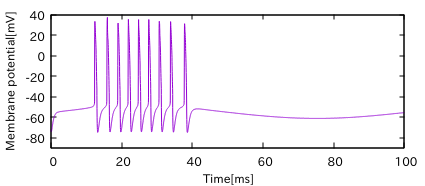


Fig.7 単精度、倍精度での実行時間の比較

(A)

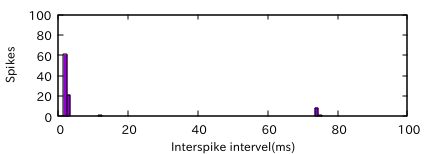
(B)

**Fig. 8 精度による膜電位変化比較**

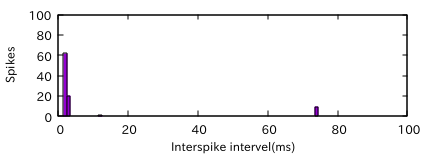
**(A) 単精度　(B)　倍精度**

1. まとめ

我々の結果から、大規模なシミュレーションではGPUの並列処理の効果は非常に大きいが、小規模なシミュレーションにおいては逆にOpenMPの方が高速と



**(A)**



**(B)**

Fig. 9 ISIによる精度比較 (A)単精度 (B)倍精度

なる現象が見られる。しかしCPU-GPU間のデータ転送が無い場合には小規模の場合でもCUDAの方が高速であることから、GPU-CPUデータ転送がこの現象の原因であると考えられる。また、CUDAにおいて単精度は倍精度と比べて計算時間の短縮が図られ、かつ精度も倍精度と変わらない。これは、GPGPUにおいて単精度計算が有効な計算手段であることを示唆する。今後は、精度の影響が大きいと思われるH-H modelのカオス領域について検証を行う。

1. 謝辞

本研究はJSPS科研費15K07146の助成を受けた。

**文 献**

1. V.K.Pallipuram, M.Bhuiyan, M.C.Smith, A comparative study of GPU programming models and architectures using neural networks, J Supercomput 61:673–718　(2012)
2. K.Fujita, Y.Kashimori, Population coding of electrosensory stimulus in receptor network, Neurocomputing 69 1206-1210 (2006)
3. 川崎雅司, “電気魚のニューロエソロジー　―混信回避行動の神経機構―, “ 科学, 岩波書店, Vol59, No7 437-445, July (1989)
4. Eugene M. Izhikevich, Simple Model of Spiking Neurons, IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 14, NO. 6 (2003)
5. A.L. Hodgkin and A.F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. J physiol, Vol. 117, No. 4, pp 500-44 (1952)
6. K.Wada and Y.Sakaguchi, Repetitive firing of a model motoneuron: Inhibitory effect of a Ca-activated potassium conductance on the slope of the frequency-current relationship, Neuroscience Research Vol. 57(2), pp 259-267 (2007)
7. B.Doiron, at el., Ghostbursting: A Novel Burst Mechanism, Journal of Computational Neuroscience 12, 5-25 (2002)
8. 藤田 一寿, 阿部祐貴, 小野寺洸哉, 樫森与志喜, GPGPUによる刺激から神経活動までを再現するシミュレーションの高速化, 社団法人電子情報通信学会、信学技報(IEICE, Technical Report) 115, 83-88 (2016)