Vol.2020-ARC-241 No.14
2020/7/31

ニューロモーフィックコンピューティングによる移動障害物を考慮した経路探索手法

櫻井 元貴[1]　穴澤 徳明[1]　上野 洋典[1]　近藤 正章[1,2]

概要：ニューロモーフィックコンピューティング (脳型計算) は，高速で省電力な計算パラダイムとして注目を集めている．本稿では，グラフ上の経路探索問題に対して脳型計算を適用し，移動する可能性のある障害物を適切に回避しつつ，速やかに目的地に到達するエージェントを検討する．所望の経路探索を達成するために，障害物の発生情報を学習しながらニューロンの閾値を調整し，該当ニューロンの発火を遅延させることで各経路の重みを変化させる．本研究の有効性を検証するために，グリッド状の経路探索問題を通じて既存手法との比較を行った．その結果，障害物の発生情報を学習して適切な経路が得られており，提案手法の優位性が確認された．

1. はじめに

経路探索アルゴリズムは自律移動ロボットの行動決定やナビゲーションサービス・自動運転など様々な分野での応用が活発に進められている．これら実問題への適用においては，計算処理の高効率化が大きな課題となる．また，経路探索アルゴリズムは通常時間的に変化しないグラフに対して用いられるが，一般の道路では経路上に渋滞が発生することや，自律移動ロボットの経路探索を行う際に経路上の障害物が移動することも考えられる．そのため，状況が時々刻々と変化するグラフに対して，変化の特徴を学習して効率的に経路探索を行う手法が今後必要になる．

本稿では，状況が変化するグラフ上の経路探索処理の高効率化に向けて，ニューロモーフィックコンピューティングの利用を検討する．ニューロモーフィックコンピューティングは脳の神経細胞を模した計算原理である．従来のノイマン型の中央集約的な処理とは異なり，ニューロンを模した各ロジックがそれぞれコアとして働き，コア間のコミュニケーションを神経活動のスパイクを模した信号で行う並列分散処理が基本となる．そのため，並列性を向上させることで特定の計算を高速に行える可能性がある．また，LSI 等での実装を考えた場合には event-driven 型で処理が行えるため，従来のデジタル回路のようにクロック同期を行う必要がなく，非同期で動作させることもできることから，電力消費が抑えられると考えられている [1].

ニューロモーフィックコンピューティングは，ニューラルネットワークを基本とする機械学習技術として応用されることが多い．ニューロモーフィックコンピューティングの特徴は，情報伝達がスパイクという単純な信号でで実現され，各ロジック上で並列分散処理が行われるということである．そのため，ニューラルネットワーク以外にも多様な応用範囲を持つ．

本稿では，ニューロモーフィックコンピューティング（スパイキングニューラルネットワーク）によりグラフの最短経路問題を解く．特に，状況が変化するような経路探索の一例として，グラフ上に通過不可能な障害物が存在し，その障害物の位置が時系列的に変化する場合について，障害物の移動情報を学習することで適切な経路を決定できる手法を検討する．我々の先行研究 [2] では，障害物が出現する確率が既知の状況で適切に行動を選択するアルゴリズムを提案した．本稿では，このアルゴリズムを発展させ，障害物の出現確率が未知のグラフにおいても適切な経路探索を行うアルゴリズムを設計する．なお，グラフの頂点と辺の数や接続関係は時間的に変化することはなく，障害物の位置に応じて各頂点の通過可能性のみが変化するものとする．

2. スパイキングニューラルネットワークの理論

近年，機械学習の分野で注目されている人工ニューラルネットワークは，神経回路を模した数理モデルである．一方，脳神経系の電気生理学的な活動をより忠実に再現した

[1] 東京大学 大学院情報理工学系研究科
Graduate School of Information Science and Technology, The Uniersity of Tokyo
[2] 理化学研究所計算科学研究センター
RIKEN Center for Computational Science

© 2020 Information Processing Society of Japan

数理モデルとして，ニューロン間の情報伝達にスパイクを用いるものをスパイキングニューラルネットワーク (Spiking Neural Network: SNN) と呼ぶ．本稿では，ニューロンのモデルとして Leaky Integrate-and-Fire ニューロン (LIF ニューロン)[3] を，シナプスのモデルとして Current-based シナプス [4] を扱う．本モデルは比較的単純なモデルであり，閉形式となるため微分数値演算が容易であることが大きな利点である [5]．

LIF ニューロンモデルは，ニューロンへの入力電流を時間積分して膜電位に反映させ，さらに時間経過で膜電位から電荷が漏れ出す仕組みを考慮したモデルである．ニューロンの膜電位を $V(t)$，静止膜電位を V_{rest}，入力電流を $I(t)$，膜抵抗を R_m，膜時定数を τ_m とすると，LIF モデルの膜電位は式 (1) の微分方程式に従う．

$$\tau_m \frac{dV(t)}{dt} = -(V(t) - V_{rest}) + R_m I(t) \quad (1)$$

膜電位 $V(t)$ が閾値 V_{th} を超えると，$V(t)$ は即座にピーク電位 V_{peak} まで上昇し発火が起こる．その後，膜電位はリセット電位 V_{reset} まで下がり，不応期 τ_{ref} の間は入力電流による影響を受けなくなる．

Current-based シナプスは，発火がシナプスに伝わった際にスパイクがそのまま電流に変換されるモデルであり，シナプスから出力される電流 $I(t)$ は，式 (2) のようにスパイク $sp(t) \in \{0, 1\}$ の定数倍で表現される．

$$I(t) = J_s \ sp(t) \quad (2)$$

ただし，J_s はゲインを示す．一般に，Current-based シナプスをはじめとするシナプスモデルは指数関数や α 関数で電流の減衰が表現され，時刻 t にスパイクが到来した場合，時刻 t の出力電流だけでなくそれ以降の時刻の出力電流にも影響を与える．本稿ではシナプスによる電流はステップ時間より短い時間で完全に減衰し，時刻 t に入力されたスパイクは時刻 t の出力電流のみに影響を与える．

ある単一ニューロンに対して，Current-based シナプスと LIF ニューロンを結合し，スパイクを入力した時の LIF ニューロンにおける膜電位の時間応答の一例を**図 1** に示す．スパイクによる電流が入力されると膜電位が増加し，時間経過によって膜電位から電荷が漏れている様子が確認できる．図の例では，30[ms] でニューロンが発火閾値である $V_{th} = 1$ に至り，リセット電位 $V_{reset} = 0$ まで下がっている．

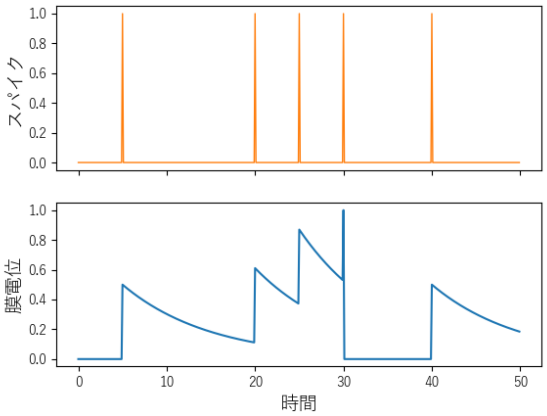


図 1 LIF ニューロンにおける膜電位の時間変化 ($\Delta t = 10^{-4}, \tau_m = 10^{-2}, \tau_{ref} = 0, V_{rest} = 0, V_{reset} = 0, V_{th} = 1, V_{peak} = 1, R = 1, J_s = 50$)

3. 先行研究

3.1 SNN による最短経路探索手法

頂点集合 V，辺集合 E で構成される無向重み付きグラフ $G(V, E)$ において，ある 2 頂点 v_{start}, v_{goal} 間の最短経路を求める問題を考える．この問題を，ニューロン集合 N とシナプス集合 S で定義される SNN にマッピングする．ここで，ニューロンの閾値を V_{th}，不応期を τ_{ref}，シナプス遅延を σ，シナプス効率を s_w とし説明する．

文献 [6] では，グラフ $G(V, E)$ を以下のように SNN へマッピングしている．まず，頂点 $v \in V$ をニューロン $n \in N$ に，2 頂点 $v_i, v_j \in V$ 間の辺 $e \in E$ を両方向のシナプス $s_{i \to j}, s_{j \to i} \in S$ に対応させる．また，全てのニューロンについて自己ループシナプス $s_{i \to i}$ を追加する．全てのニューロン n について，初期膜電位 $V_o = 0$，静止膜電位 $V_{rest} = 0$，膜抵抗 $R_m = 1$，膜時定数 $\tau_m = 1$，閾値 $V_{th} = 1$，ピーク電位 $V_{peak} = 1$，リセット電位 $V_{reset} = 0$ とする．またニューロンの不応期 τ_{ref} はステップ時間より十分小さい値とする．全てのシナプス効率を $s_w = 0$ とし，自己ループ以外のシナプスの遅延 σ は対応する元の辺の重みと等しい値にする．自己ループシナプスの遅延はステップ時間よりも十分小さい値とする．

SNN では，あるニューロンが発火するとシナプスを通じて隣接するニューロンにスパイクが伝搬する．パラメータを適切に設計することで，隣接するニューロンからのスパイクが伝搬されるとそのニューロンは即座に発火する．この性質を用いて，先行研究 [2], [6] では順探索と逆探索という 2 つの操作によって最短経路が導出されている．

3.1.1 順探索による最短時間の導出

時刻 $t = 0$ でスタートニューロンに入力電流を与え発火させる．発火はシナプスを通じて隣接するニューロンの膜電位を変化させ，発火を促す．その結果，v_{start} から v_{goal} に到達可能であれば，有限時間でゴールニューロンが発火する．ゴールニューロンが初めて発火した時刻を $t = T$ とすると，T はゴールへの最短時間を表す．

さらに，頂点と時刻の関数である頂点通過可能性 $p_f(v, t)$ を導入することで，グラフ上に動く障害物が存在する場合の最短経路問題へと拡張することができる [2]．時刻 t における頂点 $v \in V$ の通過可能性 $p_f(v, t)$ を以下のように表す．

© 2020 Information Processing Society of Japan

$$p_f(v,t) = \begin{cases} 1 \text{ (時刻 } t \text{ で頂点 } v \text{ を通過できる)} \\ 0 \text{ (上記以外)} \end{cases} \quad (3)$$

ここで，ニューロンの膜電位 $V_v(t)$ を式 (4) のように変化させる．

$$V_v(t) = V_v(t)\ p_f(v,t) \quad (4)$$

つまり，時刻 t で頂点 v が通過不可能のとき，頂点 v に対応するニューロンの膜電位 $V_v(t)$ は 0 に固定する．各イテレーションでこの更新を加えた上で，同様に n_{start} から n_{goal} までの発火が伝搬する時間を計測することで最短時間が得られる．

3.1.2 逆探索による最短経路の導出

順探索によりゴールニューロンが発火するまでの最短時間 T が求まると，以下に示す逆探索の操作で最短経路を導出することができる．まず，順探索時の時刻 t におけるニューロン n(頂点 v) の発火の有無を $\delta_f(v,t)$ で表す．時刻 t においてニューロン n が発火していた場合 $\delta_f(v,t) = 1$，発火していなかった場合 $\delta_f(v,t) = 0$ とする．

逆探索では，頂点 v の通過可能性を表す関数として以下の $p_i(v,t)$ を用いる．

$$p_i(v,t) = p_f(v, T-t)\ \delta_f(v, T-t) \quad (5)$$

この頂点通過可能性 $p_i(v,t)$ を用いて，順探索の時と同じ空間を探索する．ただし，逆探索においては v_{start} と v_{goal} の役割を入れ替える．つまり，時刻 $t = 0$ でゴールニューロンを発火し，スタートニューロンが初めて発火した段階で計測終了となる．無向グラフを仮定しているため，必ず $t = T$ でスタートニューロンが発火する．また，逆探索によって判明した，時刻 t でのニューロン n の発火の有無を $\delta_i(v,t)$ で表す．

最短経路の情報を与える関数 $l(v,t)$ は $\delta_i(v,t)$ から以下のように求められる．

$$l(v,t) = \delta_i(v, T-t) \quad (6)$$

この $l(v,t)$ は「時刻 t で $l(v,t) = 1$ ならば，頂点 v は最短時間で v_{goal} に到達する経路上の点である」ことを意味しており，最短経路の情報を与えている．

3.2 エージェントによる経路探索手法

次に，前述のグラフ $G(V,E)$ に対して，エージェントを v_{start} から v_{goal} へ到達させる問題を考える．ここでは，障害物が移動する問題ではなく，各頂点の通過可能性が変化するとして経路探索手法を考える．つまり，グラフの各頂点が時変に通過可能か不可能かが変化することになる．各頂点の通過可能性が確定している問題は，前節の手法で求めた最短経路に沿ってエージェントを動かすことで，最短時間で v_{goal} に達する．一方で，各頂点の通過可能性が未知の場合，事前に最短経路を知ることはできない．そのため，各時刻毎に現在地 v_{now} に存在するエージェントの移動先 v_{next} を決定するアルゴリズムが別途必要となる．文献 [2] では，以下の 2 つのアルゴリズムが検討されている．

3.2.1 貪欲エージェント

未来の通過可能性が未知の場合でも，現在以降の通過可能性が変化しないと仮定して最短経路を求める手法である．つまり，各時刻毎に現在のグラフの状況を基にして最短経路を求め移動先を定める．

3.2.2 確率過程利用エージェント

各時刻 t での，頂点 v の通過可能性の確率過程 $P(v,t) \in [0,1]$ が与えられている場合を考える．確率閾値 $P_{th} \in [0,1]$ を定めることにより $P(v,t)$ を二値化する．すなわち，任意の頂点 v，時刻 t に対して，$P(v,t) > P_{th}$ であれば通過可能，$P(v,t) < P_{th}$ であれば通過不可能であるとみなす．以上の判断をもとに，グラフでの最短経路を求め移動先を定める．

4. 確率過程学習エージェント

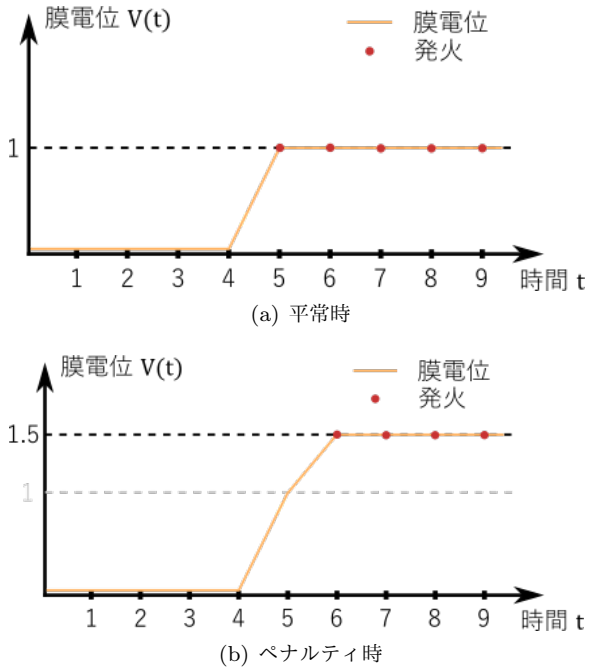
4.1 概要

貪欲エージェントは現時刻の情報のみを利用するため，最終的な経路が効率的でない可能性がある．一方，確率過程利用エージェントは確率過程が既知 (未来の情報が既知) である時のみ利用可能でるという制限がある．そこで，各頂点の通過可能性の確率過程が十分に把握できていない状況下でも，確率過程を学習して移動できるエージェントとして，確率過程学習エージェントを提案する．本エージェントは，過去データをもとに経路探索を行うものである．通過不可能になりやすいと予測される経路にはペナルティを与えることで，各頂点の確率過程を学習する．

ペナルティの設計にはニューロンの発火閾値の上昇を用いる．従来手法では，ニューロン集合全体に対して同一の閾値を設定し，隣接ニューロンからのスパイクが伝搬されると当該ニューロンは即座に発火していた．確率過程学習エージェントでは，過去データをもとに通過不可能になりやすいニューロンの閾値を上昇させることで，そのニューロンが発火する時刻を遅延させる．遅延により発火までにかかるステップ数を適切に設計することで，通過不可能になりやすいニューロンを回避するような最短経路の導出が可能となる．ニューロンの閾値の設計後は，貪欲エージェントと同様に各時刻毎に現在のグラフでの最短経路を求めて移動先を定める．

閾値の上昇によって発火が遅延する様子を**図 2(a)**, **図 2(b)** に示す．平常時は $t = 4$ で隣接するニューロンから電流が到達すると，次の時刻 $t = 5$ で即座に発火する．しかし，ペナルティとして膜電位が上昇している場合，電流が到達した直後の時刻 $t = 5$ では発火が起こらず，さらに電流が流入することで $t = 6$ で初めて発火閾値に至る．

© 2020 Information Processing Society of Japan



(a) 平常時

(b) ペナルティ時

図 2 ニューロンの閾値上昇による発火遅延

発火時刻を遅延させる具体的な実装手法については，以降で述べる．

4.2 ニューロンの閾値の設計

本手法では，グラフの現在の情報だけでなく，過去のデータ，すなわち過去の試行における情報を用いて経路探索を行う．ここでは，過去の試行における情報として過去のエージェントの位置は考慮せず，各障害物の位置のみを用いる．より具体的には，過去の時刻 t における各頂点 $v \in V$ の通過可能性 $p(v,t)$ を蓄積し，全試行についての平均をとったものを推定通過可能性 $\bar{p}(v)$ とすることで，過去データとして利用する．

$$\bar{p}(v) = \frac{1}{N_t} \sum_{t=1}^{N_t} p(v,t) \tag{7}$$

ここで，N_t はエージェントが経験した時系列データの数を示す．

次に，得られた推定通過可能性 $\bar{p}(v)$ からニューロンの閾値を設計する．頂点通過可能性が $p \in [0,1]$ であり，その頂点の通過に c ステップ要するとき，推定通過ステップ数は $\tilde{c} = c/p$ と表せる．そこで，ニューロン n (頂点 v) に初めて電流が流れ込んでから発火するまでのステップ数を $\tilde{c}$ となるように閾値を設計することで，推定通過ステップ数の総和が最小となるような経路が採択される．

4.3 膜時定数を考慮したニューロン閾値設計

LIF ニューロンモデルにおいて，膜時定数は膜電位の変化のしやすさを示すパラメータであり，“外部電流による膜電位の変化” と “時間推移による膜電位の変化” の 2 点に影響を及ぼす．本稿では，膜抵抗を無限大に近似することで膜時定数の前者への影響を無視しているが，時間推移によって膜電位の漏れは発生する．以下では，膜時定数による膜電位の漏れを考慮したニューロンの閾値設計を考察する．$\tau_m = 1$ のとき膜電位の漏れが起こらない IF (Integrated-and-fire) モデルとなるが，ここでは LIF モデルを前提に議論するため $\tau_m > 1$ と仮定する．

ニューロンの閾値 $V_{th}(v)$ を変化させたとき，そのニューロンを通過するのに必要なステップ数の増分 Δt は式 (8) で表される．ただし，膜時定数を τ_m とする．

$$\Delta t(v) = -\log_{\tau_m}(\tau_m - V_{th}(v)) \tag{8}$$

前節より，辺のコストを c とすると，ステップの増分 Δt は式 (9) のように設計できる．

$$\Delta t(v) = \frac{c}{\bar{p}(v)} - c \tag{9}$$

式 (8)，式 (9) より，各ニューロンの通過可能確率 $\bar{p}(v)$ からニューロンの閾値 V_{th} は式 (10) で計算される．

$$V_{th}(v) = \tau_m - \tau_m^{-c(1-\bar{p}(v))/\bar{p}(v)} \tag{10}$$

4.4 確率過程学習エージェントにおける最短経路探索

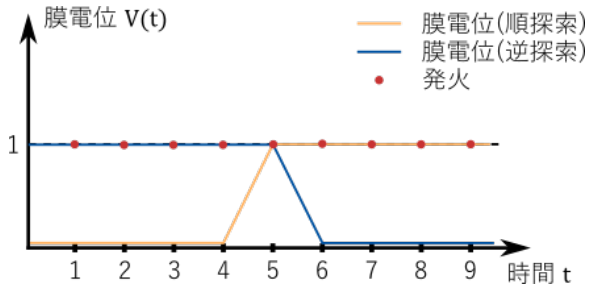
まず，3.1 節で扱った SNN による最短経路手法と同様に，所望のグラフ問題を SNN にマッピングする．ここで，ニューロンの閾値以外のパラメータについては前述のものをそのまま用いることができる．ニューロンの閾値については，前節で設計した V_{th} を各ニューロンに対して適用する．先行研究の手順と同様に，“順探索による最短時間の導出” および “逆探索による最短経路の導出” を行うことで，各ニューロンの推定通過可能確率を考慮した上での最短経路が導出できる．ただし，順探索・逆探索ともに，先行研究の手法を一部変更する必要がある．

1 点目は，順探索において，ニューロンに流れ込む複数の電流の扱いである．先行研究では，あるニューロンが発火すると隣接するニューロンは必ず次の時刻で発火していた．そのため，2 つの隣接するニューロンから同時に電流が流れ込むとき，単純に 2 つの電流の和を入力電流として問題なかった．しかし，今回の閾値を変化させる手法では，1 ステップに流れ込む電流が一定である必要がある．そのため，複数の電流が同時に流れ込む場合，入力電流はそれらの論理和にする必要がある．

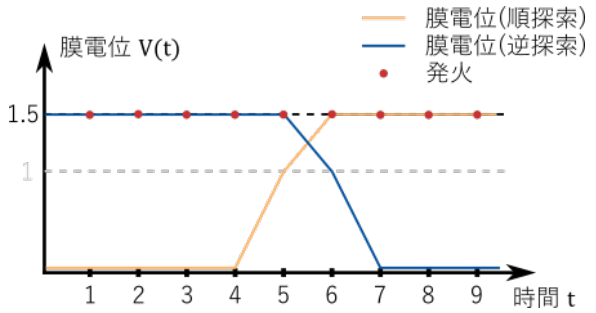
2 点目は，逆探索における頂点通過可能性 $p_i(v,t)$ の設計である．先行研究では，式 (5) に示したように，各時刻で通過可能かつ順探索の際に発火していた頂点を通過可能としていた．提案手法では式 (5) の代わりに，以下を用いる．まず，順探索時の時刻 t におけるニューロン n(頂点 v) の膜電位 $V_v(t)$ について，$\delta'_f(v,t)$ を定義する．

$$\delta'_f(v,t) = \begin{cases} 1 \ (V_v(t) > 0) \\ 0 \ (V_v(t) = 0) \end{cases} \tag{11}$$

© 2020 Information Processing Society of Japan



(a) 平常時



(b) ペナルティ時

図 3 順探索・逆探索における膜電位変化

隣接するニューロンからスパイクが 1 回でも到達しており，自身の膜電位が $V_v(t) > 0$ であった場合 $\delta'_f(v,t) = 1$ となり，隣接するニューロンからスパイクが 1 回も到達しておらず，膜電位が初期膜電位 $V_o = 0$ に等しかった場合 $\delta'_f(v,t) = 0$ となる．

この δ'_f を用いて，逆探索で用いる頂点通過可能性 $p_i(v,t)$ を次式のように設計する．

$$p_i(v,t) = p_f(v, T-t)\delta'_f(v, T-t) \tag{12}$$

なお，3 章と同様 $p_f(v,t)$ は順探索時の頂点の通過可能性を表し，T は順探索で導出された最短時間である．以上で設計された $p_i(v,t)$ を用いて逆探索を行うことで，最短経路 $l(v,t)$ を求める．ここで，**図 3(a)** のように，平常時は順探索と逆探索において同時刻で発火が起こため，順探索時に “発火していた” ニューロンを用いて逆探索を行うことで，最短経路を導出できる．しかし，**図 3(b)** のように，ペナルティ時は順探索と逆探索において発火の時刻が異なる．そこで，順探索時に”膜電位が正である”ニューロンを用いて逆探索を行う．

確率過程学習エージェントのアルゴリズムをまとめたものを**図 4** に示す．3.2 節と同様，エージェントを v_{start} から v_{goal} へ到達させるために，現在地 v_{now} に存在するエージェントの移動先 v_{next} を決定していくものである．障害物の発生位置は未知かつ動的に変化するが，各時刻で最短経路問題を解くことでエージェントを適切な経路で目的地に到達させることができる．

4.5 近傍サブグラフフィルタ

確率過程学習エージェントにおいて，障害物群の局在性を

Algorithm 1 確率過程学習エージェント

Input: $v_{now}, v_{goal}, V_{th}(v)$
Output: v_{next}
  for $t = 0\ \Delta t\ 2\Delta t \ldots$ **do**
    solve $l(v,t)$ s.t. start=v_{now} goal=v_{goal} threshold=$V_{th}(v)$
    for $\tau = 0\ \Delta t\ 2\Delta t \ldots$ **do**
      if $\max_v\{l(v,\tau)\} = 1$ **then**
        next $\Leftarrow \mathrm{argmax}_v\ l(v,t)$
      end if
    end for
  end for
  return v_{next}

図 4 確率過程学習エージェントの探索アルゴリズム

仮定することで性能を向上させることができる．Schuman ら [6] の研究により，SNN を用いて高速に近傍サブグラフ抽出が行えることが確認されている．この手法を利用し，グラフ上の各点の推定通過可能性 $\bar{p}(v)$ の値をその最近傍サブグラフ V_g に分散させることで局在性を考慮する．

$$\bar{p}'(v) = \frac{1}{N_{vg}} \sum_{v_g}^{V_g} w(v_g)\bar{p}(v_g) \tag{13}$$

ここで，頂点 v の近傍点を $v_g \in V_g$ とし，V_g の要素数を N_{vg} とする．$w(v_g)$ はサブグラフに対する重みである．つまり，各点とその最近傍サブグラフを対象にフィルタリングを行う．式 (13) で得られた新たな推定通過可能性 $\bar{p}'$ を用いて V_{th} を設計することで，過去データが十分収集できていない状況下でも適切に行動することができると考えられる．

5. シミュレーション評価

5.1 評価の仮定

前章で説明した各エージェントの性能を評価するために具体的なグラフを用いて探索実験を行う．本稿では，確率過程に従う障害物があるフィールドとして**図 5**，**図 6** の 2 つの 2 次元グリッド (15 × 30 マス) 状に規則的にノードが配置されたグラフを扱う．エージェントはこのグラフ上をなるべく早く n_{start} から n_{goal} へ移動する経路を探索する．エージェントは，自身もしくは自身の周囲 1 マス分の計 9 マスに移動可能であり，縦・横の移動には 2 ステップ，斜めの移動には 3 ステップ必要とする．グラフ上の各ノードは毎ステップ確率的に障害物が発生し，薄紫のノードは 0.9 の確率で通過可能となり，濃い紫のノードは 0.4 の確率で通過可能となる．このグラフ問題を SNN にマッピングすることにより最短経路の導出について実験を行う．SNN へマッピングする上では，各ノードにニューロンを配置し，ノード間をシナプスで結合する．

この条件下で，貪欲エージェント，$P_{th} = 0.6$ の確率過程利用エージェント，確率過程学習エージェント，および確

© 2020 Information Processing Society of Japan

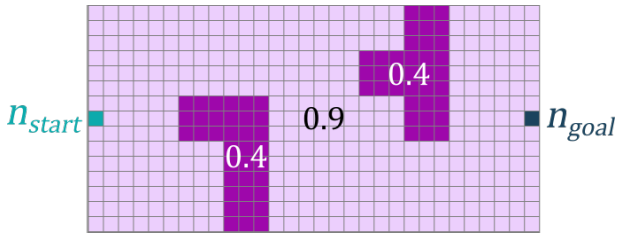


図 5 確率過程フィールド (a)

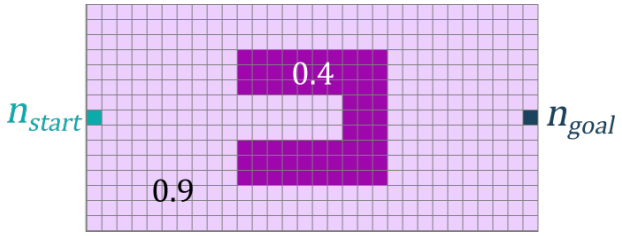


図 6 確率過程フィールド (b)

率過程学習エージェントに最近傍サブグラフフィルタを適用したエージェントの計 4 つについて，それぞれ 100 回の経路探索を行う．なお，縦方向の移動に 2 ステップ必要であり，n_{start} から n_{goal} までは最短 30 ノードがあるため，最短でも通過に 60 ステップが必要となる．

5.2 フィールド (a) の評価結果と考察

図 5 のフィールドを探索した際に，到達に要したステップ数の比較結果を**表 1** に示す．また，**図 7** に各エージェントが 100 回の探索で通過した経路の軌跡を示す．確率過程学習エージェントは平均 88.29 ステップでゴールに到達しており，貪欲エージェントの平均 92.92 ステップよりも短い時間で探索が完了している．しかし，確率過程利用エージェントでは平均 78.97 ステップとなり，確率過程学習エージェントよりも短いステップ数で到達可能となっている．一方，確率過程学習エージェントに前述の近傍サブグラフフィルタを適用することで，到達に必要なステップ数は平均 81.29 ステップとなり，確率過程利用エージェントと同程度の値になった．

以上の結果について考察する．まず，提案するの確率過程学習エージェントは貪欲エージェントよりも少ないステップ数でゴールに到達することができた．図 7 より，貪欲エージェントでは経路軌跡が広範囲に広がっているのに対し，確率過程学習エージェントでは通過可能確率が 0.4 のノード群を回避するように経路が探索されたことがわかる．通過可能確率が低いノードをうまく回避できたため，貪欲エージェントよりも早くゴールに到達できたと考えられる．一方，確率過程利用エージェントと確率過程学習エージェントを比較すると，前者の方が短いステップ数で探索が完了していることが確認できる．確率過程利用エージェントは，あらかじめ既知の各ノードの通過可能確率過程を利用するため，それが未知である提案手法よりも効率

表 1 フィールド (a) の探索結果

	貪欲	確率過程利用	確率過程学習	確率過程学習＋フィルタ
平均	92.9	79.0	88.3	81.3
標準偏差	18.3	6.04	21.5	9.97
最大値	166	95.0	203	122
最小値	69.0	66.0	70.0	70.0

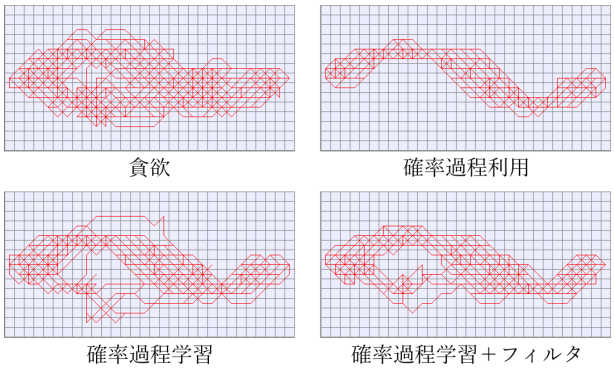


図 7 フィールド (a) の探索の経路軌跡

良く障害物を回避するような経路がとられたためである．

確率過程学習エージェントの経路軌跡から，経路後半の T 字型の障害物地帯 (通過可能確率が 0.4 のノード群) はうまく回避できているが，経路前半の L 字型の障害物地帯は効率良く回避できていないことがわかる．これは，確率過程学習エージェントは各経路探索の試行で学習データが 0 の状態からスタートするため，経路の早期に現れる障害物群はうまく回避できないことを示している．一方で，後半の障害物地帯は学習データが蓄積されているため，効果的に経路探索が行える．さらに過去の試行の経験も学習しておくことで，早期の障害物にも対応できると考えられる．

確率過程学習エージェントに対して近傍サブグラフフィルタを適用したエージェントは，確率過程学習エージェントよりも少ないステップ数でゴールに到達できている．これは，より積極的に障害物付近を回避して探索が行われるようになるため，経路前半の障害物地帯も比較的うまく回避できたことによる影響と考えられる．

5.3 フィールド (b) の評価結果と考察

図 6 のフィールドを探索した場合の，到達に要したステップ数の比較結果，および各エージェントが 100 回の探索で通過した経路の軌跡をそれぞれ**表 2** および**図 8** に示す．確率過程学習エージェントは平均 74.01 ステップでゴールに到達しており，貪欲エージェント，確率過程利用エージェントよりも短い時間で探索が完了している．一方で，近傍サブグラフフィルタを使用した場合は平均 94.31 ステップとなり，著しく結果が悪くなっている．

図 8 の経路軌跡より，確率過程利用エージェントは中央の U 字の障害物を避けるために上下に経路が分かれている様子が確認できる．一方，確率過程学習エージェントで

© 2020 Information Processing Society of Japan

表 2　フィールド (b) の探索結果

	貪欲	確率過程利用	確率過程学習	確率過程学習＋フィルタ
平均	114	79.7	74.0	94.3
標準偏差	27.9	5.69	7.63	22.3
最大値	206	93.0	92.0	192
最小値	76.0	70.0	60.0	70.0

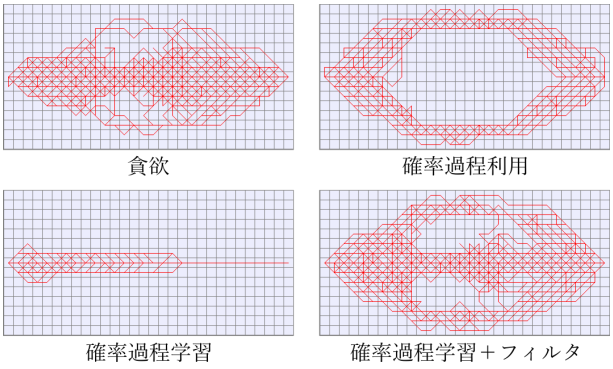


図 8　フィールド (b) の探索の経路軌跡

は障害物地帯を直進した経路がとられており，障害物地帯を直進する中央の経路と障害物地帯を回避する上下の経路の 2 つを考慮した結果，通過不可能になるリスクを冒してでも障害物地帯を直進した方がゴールへの到達が早くなると判断されたと考えられる．サブグラフフィルタを適用した場合に，確率過程学習エージェント単体の結果よりも到達に要するステップ数が増えているが，これはフィルタによって頂点の推定通過可能性の情報がぼやけ，U 字型障害物の情報を正確に捉えられなかった結果であると思われる．

6. おわりに

本研究では，グラフ上の確率過程が変化する経路探索問題について，SNN を用いて過去データを学習する手法を提案した．既存手法である確率過程利用エージェントでは事前に頂点通過可能性の確率過程が判明している時のみ利用可能であったが，提案手法では頂点通過可能性の確率過程が未知の状況でも学習によって適切な経路を選択できる．また，確率過程の局所性を仮定することで，最近傍サブグラフフィルタを用いた時に性能の向上が確認できた．

今後の課題は，あらかじめ決められた数の障害物がグラフ上を移動する問題に対して，本手法を適用することである．障害物の動きの特徴や複数の障害物の位置関係を学習することで，軌道計画にフィードバックする手法を考案したい．

謝辞　本研究の一部は，JST CREST 課題番号 JPMJCR18K1（研究課題名「エッジでの高効率なデータ解析を実現するグラフ計算基盤」）によるものである．

参考文献

[1] 森江　隆：ニューロモルフィックシステムと物理デバイス，応用物理， Vol. 88, No. 7, pp. 481–485 (2019).
[2] 穴澤徳明，上野洋典，近藤正章：スパイキングニューラルネットワークを用いた時間依存ネットワークに対する経路探索手法の検討，情報処理学会システム・アーキテクチャ研究会 2020-ARC-240 (2020).
[3] Louis, L.: Recherches quantitatives sur l’excitationelectrique des nerfs traitee comme une polarization，*Journal de Physiologie et de Pathologie Generalej*, Vol. 9, No. 620-635 (1907).
[4] Meffin, H., Burkitt, A. N. and Grayden, D. B.: An Analytical Model for the ‘Large, Fluctuating Synaptic Conductance State’ Typical of Neocortical Neurons In Vivo，*Journal of Computational Neuroscience*, Vol. 16 (2004).
[5] Cavallari, S., Panzeri, S. and Mazzoni, A.: Comparison of the dynamics of neural interactions between current-based and conductance-based integrate-and-fire recurrent networks, *Frontiers in neural circuits*, Vol. 8, No. 12 (2014).
[6] Schuman, C., Hamilton, K., Mintz, T., Adnan, M. M., Ku, B., Lim, S.-K. and Rose, G.: Shortest Path and Neighborhood Subgraph Extraction on a Spiking Memristive Neuromorphic Implementation, pp. 1–6 (online), DOI: 10.1145/3320288.3320290 (2019).

© 2020 Information Processing Society of Japan