Vol.2021-MPS-132 No.20
2021/3/2

ローグライクゲームにおける多目的神経進化に基づくモジュラーネットワークの導入

高橋　寿徳[1]　棟朝　雅晴[2]

概要：機械学習を現実社会に応用するには，学習環境の状況に応じた柔軟な学習が行えるのが望ましい. 本研究では，ローグライクゲームをテストベッドとして，次の二つの課題に取り組む.(1) 参照点に基づく多目的神経進化の考え方を導入し，一度の学習で複雑な問題の解決に寄与する多様な戦略を持ったニューラルネットワークを作成し，従来の多目的神経進化の手法と比較する.(2) 多目的神経進化で作成したネットワーク群をモジュラーネットワークとして活用し，ローグライクゲーム中の現階層等の情報に応じて戦略の切り替えを可能とするニューラルネットワークの実現を目指す.

Introducing modular networks based on multi-objective neuroevolution in rogue-like game

1. はじめに

現実社会の問題において、最善となる解決策が一つに決まらない場合や、情勢の変化によって従来の解決策では効果が無くなることは少なくない.

本研究では，多目的神経進化の考え方を導入し，一度の学習で多様な戦略を持ったニューラルネットワーク群を作成し，そのニューラルネットワーク群を応用することを目標とする. そのテストベッドとして，ローグライクゲームを複数の戦略が獲得できる学習題材として捉え，参照点を用いた多目的最適化の考え方を導入した神経進化を導入し, 従来の多目的最適化と神経進化を比較して多様性のある最適解群が得られるかを評価する.

さらに，取得した解群をモジュラーネットワークとして利用し，ローグライクゲームの攻略において複数の戦略の組み合わせが有効かどうかについて調べる.

2. ローグライクゲームの戦略学習

Rogue を自動プレイする研究の一つとして Michael らによる ROG-O-MATIC[1] が 1984 年に発表されており，Rogue で取るべき行動をルールベースで選択するのを突き詰めた研究であるといえる.

近年ローグライクゲームに機械学習を導入した研究として,Kanagawa らによる研究 [2] と,Andrea らによる研究 [3] の二つが発表された. これらの研究は, 共に強化学習の手法 [4][5] を利用してフロアを探索し, 既に通ったことがある場所の情報や, 自機周囲のオブジェクトの種類を入力として与え, 未探索の部分を優先的に探索しつつ, 階段を見つけたら降りて次の階層に進むことが達成できた.

その一方で, これらの研究で達成できなかった点として, なるべく未探索の部分を埋めてから階段を下りるという目標と比較すると, 階段を見つけ次第即座に階段を降りるという, 期待通りではない学習結果になったことが示唆されている.

3. 多目的最適化と神経進化の融合

3.1 神経進化

神経進化はニューラルネットワークを個体, ネットワーク構造を個体の遺伝子として扱った進化計算手法の一つである. 神経進化で代表的な手法として NEAT(NeuroEvolution of Augmenting Topologies)[8] が挙げられ,Kenneth らは個体としてのニューラルネットワークの扱い方について，

(1) 履歴マーカーによる遺伝子追跡.

(2) 種集団への分化による多様性の保護.

1 北海道大学大学院情報科学院
Graduate school, of Information Science and Technology, Hokkaido University.
2 北海道大学情報基盤センター
Information initiative center, Hokkaido university

ⓒ 2021 Information Processing Society of Japan

(3) 小規模なネットワークからの増分進化.
の導入によって, ネットワークを直接進化計算で扱うより高精度で堅牢性のある最適化を可能とした.

3.2 提案手法

3.2.1 RNSGA2-NEAT における参照点に基づいた種集団中の個体群の優劣決定

本研究では多目的神経進化の手法として, 参照点の考え方を導入した NSGA2 の派生系の RNSGA2[9] を基とした個体選択手法を取り入れ, 種集団の探索空間を広げることに焦点を当てた RNSGA2-NEAT を採用する.

この手法ではパレートランクと参照点からの距離に基づいて生き残り選択を行う, 提案手法の新規要素を赤色のノードで表現すると, 提案手法のフローチャートは以下のようになる.

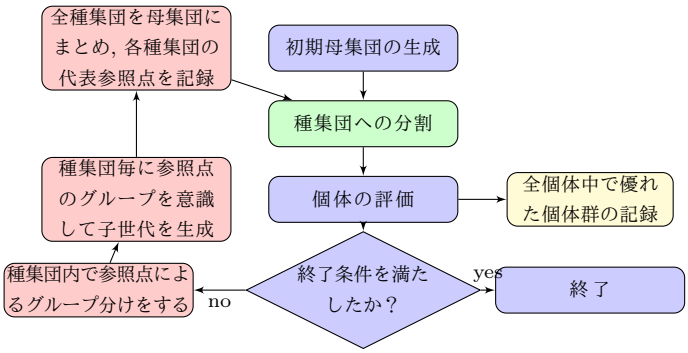

図 1: 提案手法の処理手順

フローチャートで示されているように, 種集団内で参照点を計算し, その参照点を基にグループ分けすることで, 適応度が類似した個体同士でのみ交叉が行えるように仕向ける.

参照点は [10] と同様の手順で，種集団内の極点を目的関数の数だけ求めて参照点として利用し，個体群は各参照点について最短の参照点を取ったとき，より参照点に近い個体である程優れた個体とみなす.

なお，参照点からの近さはパレートフロントが入りきらないランクの個体群を選択するのに使用する. ただし，近傍個体との距離が ϵ 以下となる混雑した個体群は混雑距離と同様の考え方でペナルティが与えられる.

3.2.2 参照点による分割と, 大域的探索の両立について

事前実験では，種集団中の個体群を参照点に基づいて分割した後, 各グループの個体数が等しくならない場合が多く見られた. そのためこのままグループ毎に交叉, 突然変異を行うだけでは, 種集団の中で探索が局所的になるという懸念がある.

そこで, 種集団を参照点によって分割する際, 種集団の代表的な参照点を記録して，これまでに注目した参照点履歴という全域的な情報を提示し，未知の解に種集団が接近するための手掛かりとした.

参照点でグループ分けを行った後の処理について図 2 に示す.

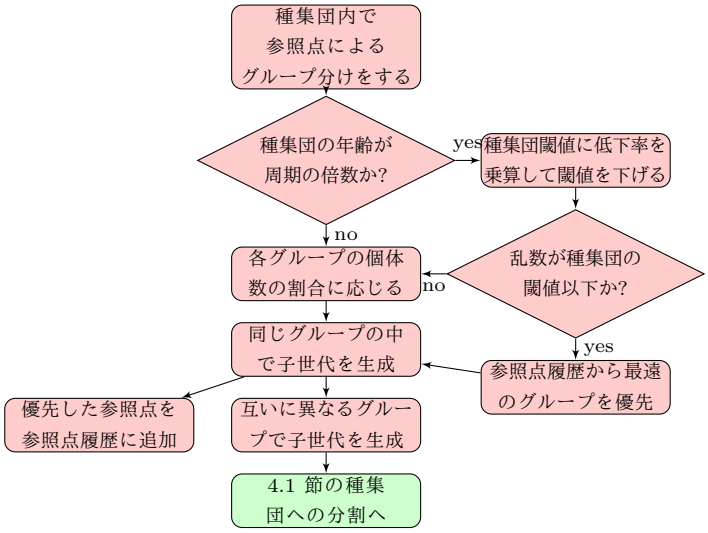

図 2: 参照点によるグループ分け後の処理手順

3.2.3 学習後のネットワーク出力について

学習が進むにつれて徐々に優秀な個体群が確保されるようにするべく, 図 3 に従って毎世代個体保存を行う.

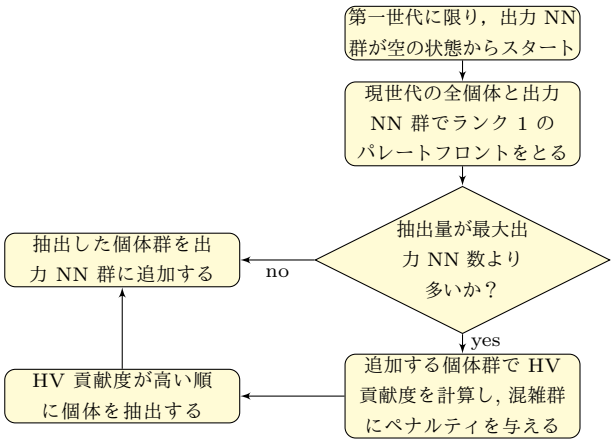

図 3: 個体群記録のフローチャート

4. 学習環境と設定について

4.1 Rogue を学習するネットワークについて

本研究において, 多目的神経進化でローグライクゲームを学習するネットワーク構造は図 4 の通りとする.

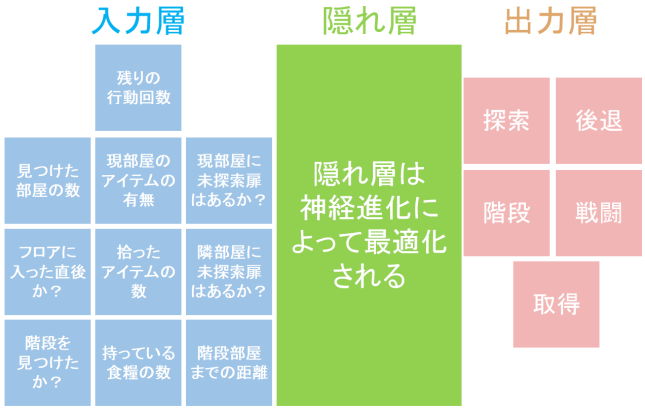

図 4: ローグライクゲーム学習に使用するニューラルネットワークの構造

出力は以下に示す一連の行動から選ばれる.

(1) 現在の部屋の扉から未探索の扉を通り，別の部屋に移動する.(探索)

(2) 直前にいた部屋に戻る.(後退)

(3) 階段がある部屋に近づくように移動する/部屋に階段がある場合は階段を降りる.(階段)

(4) 敵を引き付け，近接戦闘を仕掛ける.(戦闘)

(5) 部屋の中のアイテムを拾いにいく.(取得)

また, 評価関数として, 次の評価基準を採用する. なお, これらの評価基準は 1 フロアを 10 回挑戦した際の平均で表され,0 から 1 までの範囲に収まるよう正規化して取り扱う.

(1) 階段を降りるのに成功した割合

(2) 見つけた部屋の数の平均

(3) アイテムを拾った回数の平均

5. 多目的神経進化の実験と結果

本研究ではローグライクゲームの攻略に寄与し, 多様性のあるネットワーク群を出力するのが目標である. そこで, 評価指標として, 出力ネットワーク群の超体積 (HyperVolume:HV) を指標とする. HV は以下の式によって計算され, ある個体 a による $vol(a, z^*)$ は, ある個体 a の多次元適応度と鉛直点 z^* 間の体積によって計算される.

$$HV(A) = HV(A, z^*) = \bigcup_{a \in A} vol(a, z^*) \qquad (1)$$

式 1 で,A を個体群,a を A 中のある個体,z^* を鉛直点とする. 今回, 鉛直点は (1.1,1.1,1.1) と設定した.

高い HV を示す個体群は，優秀な多目的最適解を持ち，解に多様性があると捉えられる. 提案手法にと他の多目的神経進化を比較し，30 回試行した結果について図 7 から図 9 に表す. これらの比較手法について，個体選択の基準等は表 1 に表す.

表 1: 比較する各多目的神経進化の個体選択・交叉の仕様

名前	個体選択基準	交叉相手の決定
NSGA2	非優越ソート+混雑距離	生存個体群から乱択
PS	支配個体のパレート強度の合計	生存個体群から乱択
NSGA2HV	非優越ソート+混雑距離+HV 指標	生存個体群から乱択
NSGA3	非優越ソート+参照線による分散化	生存個体群から乱択
RNSGA2	非優越ソート+参照点からの距離+混雑距離	グループ内でのみ交叉

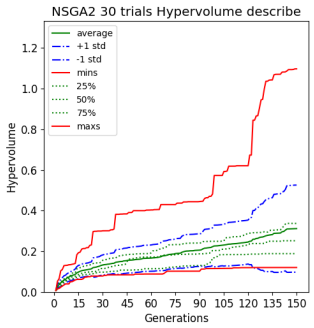

図 5: NEAT-NSGA2 における HV の遷移

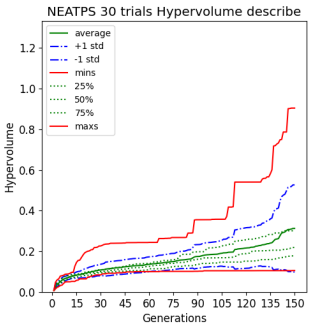

図 6: NEAT-PS における HV の遷移

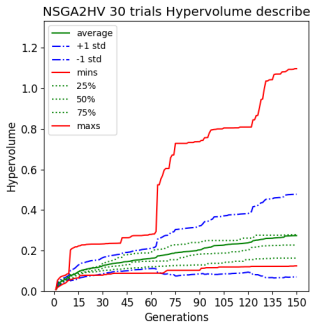

図 7: NEAT-NSGA2-HV における HV の遷移

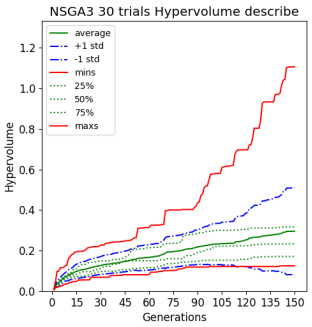

図 8: NEAT-NSGA3 における HV の遷移

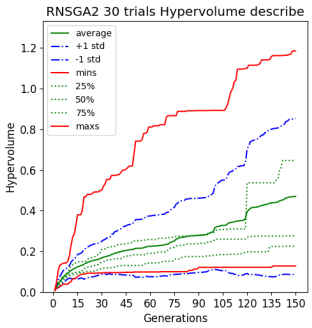

図 9: 提案手法における HV の遷移

上記で列挙したグラフから，平均, 最大で比較すれば提案手法 (NEAT-RNSGA2) が最も優れている一方, 分散が非常に大きいという点から, 安定して高い HV を出力するにはまだ改善の余地があるといえるが，一つ目の目標における，多様性のある解群を出力することについて一定の成果を出せたといえる.

6. 多目的神経進化後のネットワーク群の応用について

6.1 モジュラーネットワーク導入のアーキテクチャ

二つ目の目標に対応するべく，提案手法で出力したネットワーク群をモジュラーネットワークとして応用する．

ここで，30 回試行の中で出力されたネットワーク群全てからモジュラーネットワークとして活用が見込める群を抽出する.RNSGA2 と同様に 3 つの参照点を (1.0,1.0,1.0),(0, 1.0, 1.0), (1.0, 1.0, 0) として，各参照点への最短距離に基づいて 3 つのグループに分け，グループ毎に，パレートフロントランクと混雑距離を基準として上位 15 個体を抽出すると図 10 のようになる．

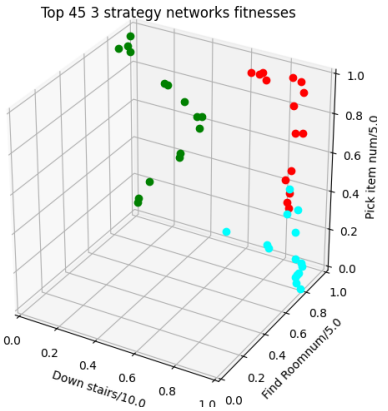

図 10: 各戦略毎に NSGA2 と同様に上位 15 個体を抽出したときの適応度分布

また，戦略毎の 15 個体はアンサンブルし，次の行動を多数決によって決定する. 応用時の学習に使用するニューラルネットワークのアーキテクチャの一例を図 11 に示す．

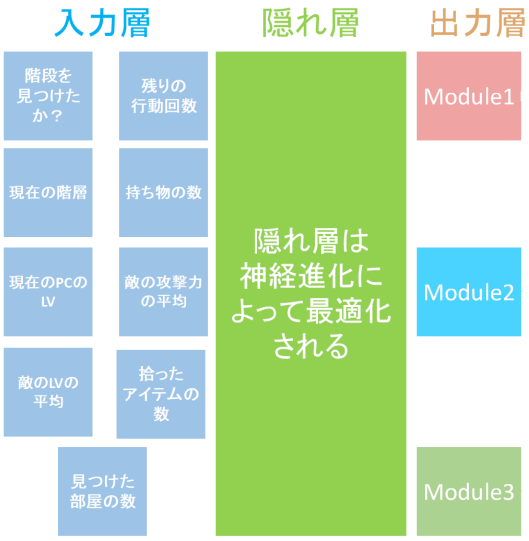

図 11: 3 つの戦略を応用する際のネットワーク構造

このネットワークに入力を与えるのも第 4 節と同様に階段を降りた直後と部屋に入った直後としている．

6.2 応用時の学習環境について

この応用では 1 フロアだけではなく，一定階層まで階段を降り続けることを想定し，途中で PC が倒されるか，地下 20 階層を超えて階段を降りた場合に 1 試行の終了とする．

ただし，学習時に考慮していなかった箇所 (敵との戦闘やアイテムの使い方について) があるため，学習時と比較して次のような緩和を行っている．

(1) 不適切な行動を選択したとき，その回の試行を打ち切らずに実行可能な代替行動 (探索, 後退など) を行う．
(2) 部屋または通路で敵と遭遇した際は，ルールベースによる戦闘行動を行う．
(3) アイテムの使い方も必要に応じてルールベースに従って決定する．
(4) 1 フロアの滞在時間制限として，それまでの行動回数が 20 回を超えたら，階段へ向かう行動を取り続ける．

6.3 応用時の学習設定について

応用時の NEAT アルゴリズムでは，目的関数は一つだけとし，1 個体につき 5 回の試行を行ったときの最大到達階層の平均とする．

ここで，図 11 において Module1 を balance，Module2 を stairs，Module3 を item と呼称したとき，比較するモジュールの組み合わせは表 2 の通りとなる．

表 2: 比較するモジュラーネットワークの一覧

モジュールの組み合わせ	備考
balance + stairs + item	単目的 NEAT で学習を行う
balance + stairs	単目的 NEAT で学習を行う
balance + item	単目的 NEAT で学習を行う
stairs + item	単目的 NEAT で学習を行う
balance 単独	学習を行わない
stairs 単独	学習を行わない
item 単独	学習を行わない

6.4 本番環境挑戦時の動向について

単目的 NEAT で学習した四通りの複数モジュール導入ネットワークと，未学習の単一モジュールによって学習時と同様の環境で 50 回試行したときの結果をまとめる．

複数モジュール導入ネットワークでは，(1) 各階層の累計踏破率と，(2) 使用したモジュールの内訳の二つを調べ，単一モジュールでは，到達階層のみを調べる．階層の踏破率とは，各階層毎に階段を降りて次の階層に進めた割合を表す．

複数モジュール導入ネットワークでは，10 世代目から 20 世代目の間に母集団の中で最大の適応度を示した個体を 10 個体抽出するのを 5 回行って，50 個の学習後のネットワークを用意し,1 つのネットワークにつき 1 回試行した結果で累計をとる．

6.5 単一モジュール

一つのモジュールのみを用いて 50 回テストを行ったときの各階層の踏破率を図 12 から図 14 に示す.

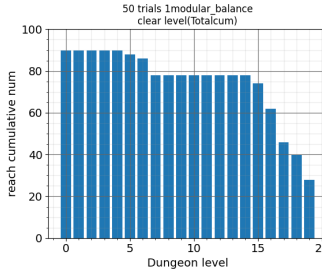

図 12: balance モジュール単独による累計階層踏破率

図 13: stairs モジュール単独による累計階層踏破率

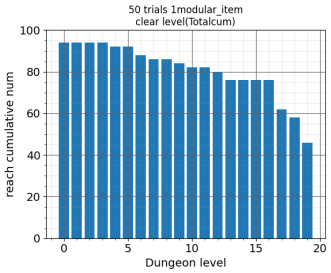

図 14: item モジュール単独による累計階層踏破率

この結果から伺えるように，深層へ潜るにはアイテムを拾う行動を習得しているかが重要であるといえる.

6.6 複数モジュール導入

学習後の複数モジュール導入ネットワークにおける各階層の踏破率を図 15 から図 18 に示す.

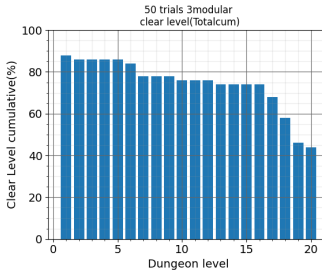

図 15: 3 つのモジュール導入時の累計階層踏破率

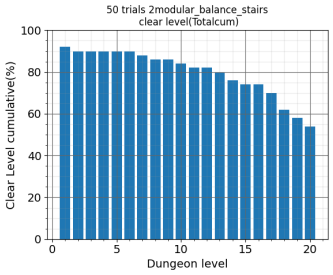

図 16: balance と stairs 導入時の累計階層踏破率

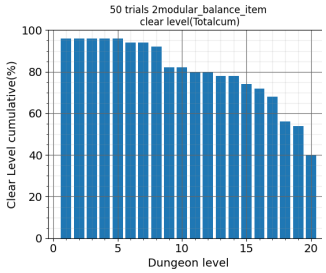

図 17: balance と item 導入時の累計階層踏破率

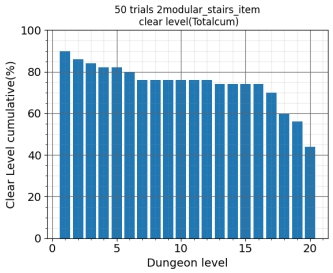

図 18: stairs と item 導入時の累計階層踏破率

これらの図は最深層までの到達率が全て 40%を超えており，アイテムを拾う戦略は複数モジュール導入時も重視されているといえる. 次に, 階段発見前後における各モジュールの使用率についてを図 19 から図 26 に示す.

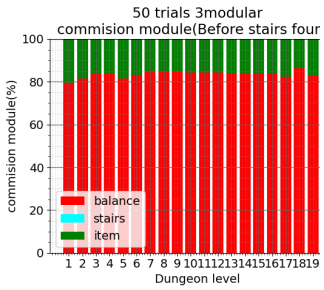

図 19: 3 つのモジュール導入時のモジュール使用率 (階段発見前)

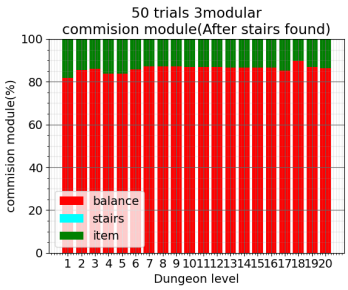

図 20: 3 つのモジュール導入時のモジュール使用率 (階段発見後)

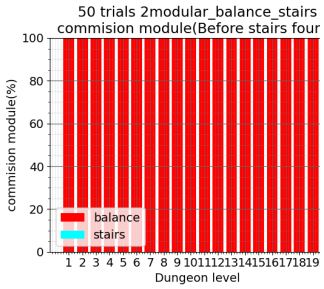

図 21: balance と stairs 導入時のモジュール使用率 (階段発見前)

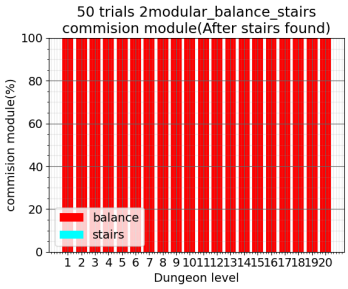

図 22: balance と stairs 導入時のモジュール使用率 (階段発見後)

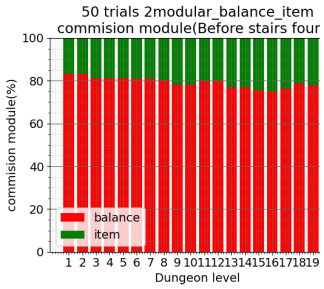

図 23: balance と item 導入時のモジュール使用率 (階段発見前)

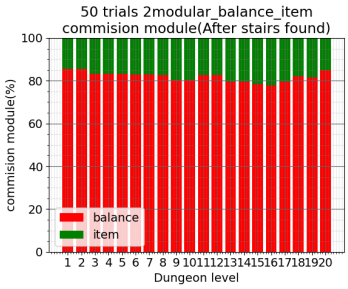

図 24: balance と item 導入時のモジュール使用率 (階段発見後)

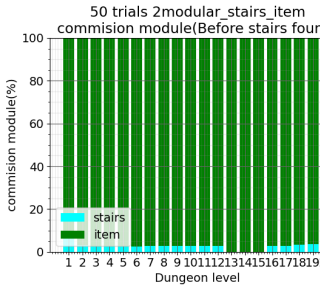

図 25: stairs と item 導入時のモジュール使用率 (階段発見前)

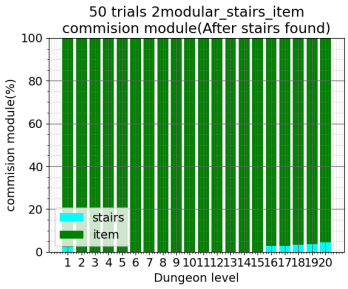

図 26: stairs と item 導入時のモジュール使用率 (階段発見後)

モジュール使用率で示唆されるモジュールの使用傾向について以下の三つが挙げられる.

(1) 階段の発見前後で使用するモジュールは大きくは変わらない
(2) stairs モジュールは深層への到達に殆ど活用されていない.
(3) 未探索部屋への移動，アイテムの取得を漏れなく行うモジュールである程深層への到達に貢献する.

これらの傾向から，本研究における応用時の学習環境では緻密に探索を行うモジュールを用いる方が深層まで辿り着く可能性が高く，balance モジュールと item モジュールが深層に到達する戦略として多く使用されていた.

ただし，この結果は第一章で述べた，複数戦略を切り替えながら，安定して深層まで到達できるようにする目標には反した結果となっている. このような結果になった理由は二つ考えられる.

(1) アイテムを多く拾って PC の強化や一時凌ぎの手段の収集に勤しむほど強敵が多い深層においても長く探索する余裕が生じていた.
(2) stairs モジュールがアイテムを一切拾わない一方で，多くの部屋を見つけるように行動するモジュールとして機能しており，階段を即座に降りる戦略として機能していなかった.

残念ながら本研究の範囲では複数モジュールの活用によって学習が効率化されることまでは十分に示せなかったものの，モジュールの抽出方法や，学習環境について設定を変えることで，異なる結果となる可能性がある.

7. 結論

本論文では，参照点を導入した多目的神経進化でローグライクゲームを学習し，ネットワーク構造に加えて適応度が近い個体同士を交叉させることで，従来の多目的神経進化と比較して，適応度のバランスに優れた個体群の抽出が可能となるのを示唆した.

ただし，モジュラーネットワークの設計では，学習環境の設定や目的関数による影響として，一部の行動だけ使用したり，特定の戦略へ収束しないなどの現象が起きたことから，真に多様な戦略を獲得できたとは言い難い結果も得られた. そのため，これらに対処するために各出力に対応するよう目的関数を増やすなど，さらに粒度を上げた解析も必要になると考えられる.

また，多目的神経進化で適応度の傾向が異なる個体群を複数抽出し，モジュラーネットワークとして利用した結果では，一つのモジュールだけでローグライクゲームに挑戦するよりも安定して深層へ到達する複数モジュールの組み合わせが一部見られたものの，組み合わせによっては一つのモジュールのみしか利用されない場合もあるなど，モジュール同士の優劣が発生する一面もあり，モジュールの抽出方法や学習環境について他にも様々な設定で試す必要性も感じられた.

今後の課題として，3 目的以上の問題として目的関数を増やした場合などでも提案した手法が役に立つのかと，ローグライクゲーム以外の環境でも成果が出せるかについても調査が必要になる.

8. おわりに

本研究は JSPS 科研究費 JP20K11967 の助成を受けたものである.

参考文献

[1] Michael L. Mauldin, Guy Jacobson, Andrew Appel and Leonard Hamey.: *ROG-O-MATIC: A Belligerent Expert System*, Fifth Biennial Conference of the Canadian Society for Computational Studies of Intelligence 入手先 〈https://www.cs.princeton.edu/ appel/papers/rogomatic.html〉

[2] Yuji Kanagawa ; Tomoyuki Kaneko.: *Rogue-Gym: A New Challenge for Generalization in Reinforcement Learning*, Proc of 2019 IEEE Conference on Games(CoG).

[3] Andrea Asperti ; Daniele Cortesi ; Carlo de Pieri ; Gianmaria Pedrini ; Francesco Sovrano.: *Crawling in Rogue' s dungeons with deep reinforcement techniques*, IEEE Transactions on Games Vol.12, pp 177-186(2019).

[4] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov.: *Proximal Policy Optimization Algorithms*, OpenAI(2017).

[5] Ziyu Wang,Victor Bapst,Nicolas Heess,Volodymyr Mnih,Remi Munos,Koray Kavukcuoglu,Nando de Freitas.: *SAMPLE EFFICIENT ACTOR-CRITIC WITH EXPERIENCE REPLAY*, DeepMind(2017)

[6] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, T. Meyarivan.: *A fast and elitist multiobjective genetic algorithm: NSGA-II*, IEEE Transactions on Evolutionary Computation Vol.6 pp. 182-197 (2002).

[7] Eckart Zitzler,Marco Laumanns.: *SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization*, EUROGEN'2001.

[8] Kenneth O. Stanley, Risto Miikkulainen. : *Evolving Neural Networks through Augmenting Topologies* Evolutionary Computation 10(2): pp. 99-127(2002).

[9] Kalyanmoy Deb, J. Sundar. : *Reference Point Based Multi-Objective Optimization Using Evolutionary Algorithms* GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation. pp 635-642(2006).

[10] Julian Blank, Kalyanmoy Deb, and Proteek Chandan Roy. : *Investigating the Normalization Procedure of NSGA-III* Evolutionary Multi-Criterion Optimization. EMO 2019. Lecture Notes in Computer Science, Vol 11411. pp 229-240 (2019).

[11] Steven Kunzel,Silja Meyer-Nieberg.: *Coping with opponents: multi-objective evolutionary neural networks for fighting games*, Neural Computing and Applications Vol.32, pp 13885–13916(2020).