5P-07

# 複雑なタワーディフェンスゲームのAI開発を目的としたシミュレータ開発

野村　大輔†　　秋岡　明香‡

明治大学 総合数理学部

## 1. はじめに

タワーディフェンスゲームはリアルタイムストラテジーゲームの一種で作品ごとに差異はあるものの、一般に、入口から登場する敵を目的地に到達させないように味方ユニットであるタワーを建設し、敵を妨害、撃破するゲームである。現在、このタワーディフェンスゲームの種類は多岐にわたり、純粋な、タワーを建設しそれを強化することによって敵の進行を阻止するタイプのタワーディフェンスゲームから、所持しているキャラクターを強化し、各タワーの持つスキルを駆使して攻略する複雑な要素の存在するタワーディフェンスゲームまで存在する。純粋なタワーディフェンスゲームのシミュレーションはいくつか存在するが複雑なタワーディフェンスゲームのシミュレーションは少なく、さらに、強化学習を目的としたシミュレーションは利用することが難しい。そのため、本研究では複雑なタワーディフェンスでのAI開発を目的としたシミュレータを作成した。

## 2. 関連研究

複雑なタワーディフェンスのシミュレータを作成し、ステージ生成とステージを解くAIをそれぞれ作成する研究が存在する[1]。本研究はあらかじめ入力したステージのみを学習に利用するのに対し、先行研究では、AIの生成したステージも用いられる。また、先行研究では利用できるタワーの種類は12種類のみであり、エピソードごとにキャラクターを選択することは無いという点で異なっている。

## 3. シミュレータ概要

本研究ではUnityのML-Agentsでの強化学習を目的とし、現在人気なソーシャルネットワークゲームのルールをもとにシミュレータを開発した。図1に開発したシミュレータの概要を示す。シミュレータはBrain、Brain_sub、Start、Enemy、Towerを中心に構成される。

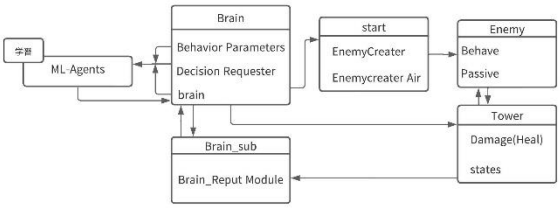


図1 各オブジェクト関連図

Brainはこのシミュレータの心臓であり、Behavior Parameter、Decision Requester、Brainを持つ。Behavior Parameterは盤面情報の観測長や推論モデルが決定するアクションの範囲などを持ち、推論に用いるモデルもここに持つ。Decision Requesterは一定フレームごとにBehavior Parameterが持つ推論モデルへ決定を要求する。BrainはAgentスクリプトであり、エピソード開始時にステージ、Start、Brain_subを生成する。また、エピソード中推論モデルの収集する盤面情報の指定やその情報から決定された行動に沿ってタワーの生成、破壊、スキルの発動を指定する。さらに、エピソード終了時にモデルの更新に用いる報酬もここで設定する。

Brain_subはBrainをエピソードごとに作りなおすことができないため、代わりにエピソード開始時に生成されるオブジェクトである。Brain_Reput Modeuleを持ち、これは各タワーが持つ破壊時に再配置が可能になるまでの時間の管理のためのスクリプトで、Brainでタワーの破壊が指定された際と一部タワーが持つスキル終了時に自身を破壊する際に呼び出され、敵ユニットにタワーが破壊された際はTowerのStatesを通して呼び出される。

Startは敵ユニットの生成位置であり、EnemyCreaterを持つStartとEnemyCreater Airを持つAir Startの2種類が存在する。EnemyCreaterは敵ユニットで

ある Enemy を生成するスクリプトであり、敵の生成のタイミングや敵の種類、動き方は Brain が Start を生成するときに指定する。EnemyCreater Air は基本的な機能は EnemyCreater と同じであり空中の敵ユニットの生成を担当する。

Enemy は敵ユニットであり、行動をつかさどる Behave とステータスを管理する Passive を持つ。Behave は Enemy の移動を制御し、また自ユニットへの攻撃もここで管理する。Passive は Enemy の攻撃力、体力、防御力、移動速度などのステータスを持ち、自タワーから攻撃を受ける際や、自タワーへ攻撃を与える際のダメージ量の決定などの際に呼び出され、体力が 0 になった際はこのスクリプトの機能によってユニットが破壊される。状態異常もこのスクリプトで管理しており、自ユニットのスキルによって状態異常が発生した際も呼び出される。

Tower は自ユニットであり、Damage または Heal と States を持つ。Damage は敵ユニットへの攻撃を制御し、敵の取得、攻撃、クールタイムを繰り返す。各キャラクターの持つスキルも関数としてここに格納され、Brain からの指示に従って発動する。States はタワーの体力などのステータスを管理し、体力がなくなれば Brain_Reput Module を呼び出す。

エピソードを開始すると、Brain によって設定したリストからランダムにステージが選択され、そのステージに合わせて Wall（壁）、Start（敵出現位置）、Goal（自拠点）が生成される。ステージが生成されると、EnemyCreater によってステージごとに決められた Enemy（敵ユニット）が生成され、各敵ユニットは自拠点へ向かってステージごと決められたルートで侵攻してくる。

エピソード中は Decision Requester で指定したフレームごとに Brain に含まれる Sensor で観測した盤面の情報が推論モデルに送られ、そのモデルから Brain に行動が intList で出力され、その値をもとにタワーの設置や破壊、スキルの発動を行う。今回の入力は固定長観測として、マップ情報配列、選択した自ユニット、自ユニットのスキルの発動可否、自ユニットの残存体力、その他所持コストなどの情報を持ち、可変長観測として、自拠点の位置と敵拠点の位置、盤面上の敵の位置とその種別が取得されている。出力は Discrete Action が長さ 4 で設定されて、Branch0 は自ユニット設置時の座標指定、Branch1 は自ユニット設置時の向き指定、Branch2 は自ユニット設置時やスキル発動、タワー破壊時の自ユニットの種類の指定、Branch3 はスキルの発動と自ユニットの破壊を担当している。

Tower（自ユニット）は敵に攻撃可能な距離によって分かれており、遠距離、近距離、超近距離、自ユニットへ回復の 4 種類に分かれており、さらに遠距離 10 種、近距離 13 種、超近距離 4 種、回復 4 種のユニットが設定されており、それぞれ異なるステータス、スキルが設定されており、キャラによっては攻撃できない敵ユニットが存在する。特に、EnemyCreater Air から生成される空中の敵は遠距離ユニットと一部近距離ユニットによってのみ攻撃が可能である。

各自ユニットはステータスが異なるのと同様にそれぞれ設置に対しコストが設定されており、設置する際にそのコストを払って設置する。このコストは時間で増加するだけでなく、一部のユニットのスキルを発動した際にも増加し、また自ユニットを Brain の指示の下破壊した場合設置時のコストの一部が返還される。

一度のエピソードでは 12 種までタワー選択することができ、同一種のタワーは同時に置くことができない。また、タワーが破壊されると Brain_Reput Module が呼び出されキャラごとに指定された時間が経過した後再配置が可能となる。

エピソードは、決められた全ての敵ユニットが撃破される、または自拠点へと到達すると終了し、モデルの学習中では、Brain から報酬が与えられモデルを更新する。また、新しくステージを生成するため、すべての壁、敵出現位置、自拠点、自ユニットは一度破壊され、次のエピソードが始まり、再度ランダムにステージが選択され壁などが設置される。

## 4. おわりに

今回は複雑なタワーディフェンスを想定してシミュレータを作成したが、まだまだシミュレータ上での敵の種類やスキルの種類が少なく、スキルも想定の挙動とは異なっているため、改善の余地が存在する。また、モデルに関連する収集する盤面の情報や出力の各 Branch は調整が不十分であったため、今後はより良い強化学習の環境を目指して調整していきたい。

### 参考文献

[1] Y. Xu and T. Tanaka, “Procedural content generation for tower defense games:a preliminary experiment with reinforcement learning” 第 26 回ゲームプログラミングワークショップ（GPW-21）