ゲーム AI プログラミング大会用コード作成

谷聖一研究室　新井 悠太
Yuta Arai

概要

人工知能 (Artificial Intelligence, AI) やそこから派生した技術は，幅広い分野に応用されている．そこで，若い世代の人材育成を目的とした AI プログラミングコンテストが開催されている. そのようなコンテストの 1 つである，SamurAI Coding のコード作成し，またマリオ AI のコードを試作した.

1　はじめに

知能とは実際の目標を達成する能力の一部であり，判断，理解，推理，計算，学習など様々なものがある．人工知能 (Artificial Intelligence, AI) とは簡単に言うと，コンピューターに知的な活動をさせることを目的とする研究と技術のことである．しかし，この知的な活動とは何か，知能があるかとはどのようなことか自体に大きな問題あり，立場により人工知能の定義も異なっている．

人間の知的な活動をモデル化すると科学的側面と工学的側面があると考えられる ([1]).

科学的側面 人間がどのように問題を解決しているのか，人間のやり方に沿ったモデルを構築し，コンピュータ・シミュレーション等を行うことによって知識の原理，人間の思考過程を解明しようという側面.

工学的側面 人間がどのようにやっているかということには拘らず，とにかくその「知識を必要とすると思われる問題」を解明できる知的情報処理システムを実現しようという側面.

注意として，この２つの側面は明確には区別されていない．現在では後者の研究がより広く進んでおり，人工知能から派生して独立した分野になった，自然言語処理，パターン認識，エージェント学習，データマイニングなどがある．ゲーム AI についても工学的側面の知的情報処理を中心とした研究となっている.

報告者は，ゲーム AI について理解を深めることを目的に SamurAI Coding というゲーム AI プログラミングコンテストに挑戦した. SamurAI Coding([2]) では主にエージェントの行き先を決定する priorityMove 関数を作成し，それを中心にプログラムを作成した. 稿では SamurAI Coding の予選，決勝の時の戦略，またそれがどのように有効であったかと，敵チームの戦略，その対応について述べる. 又マリオ AI([3]) についても試作した.

2　SamurAI Coding 2013 について

2.1　概要

SamurAI Coding は 2011 年度に早稲田大学鷲崎研究室が開催し，2012 年度からは情報処理学会が継承して開催している. 大会のウェブによると若い世代から将来第一線の研究者や開発者になりうる，また世界市場を舞台に活躍できる人材を育てることを目的とした大規模な国際的なコンテストである. また今年は東京大学大学院情報理工学系研究科が決勝で共催し，GREE 株式会社がイベントスポンサー，その他ゴールドスポンサー，協賛，後援の方々が支援している ([2]).

ゲームシステムは簡単に説明すると，六角形のセルが敷き詰められた盤面をそれぞれ侍エージェントと犬エージェントが 1 ターンに 1 回行動し，領土を広げてポイントを競うゲームである．昨年とゲームシステムはあまり変わらないが，プログラム記述方法が変更されたため，昨年度のプログラムソースをそのまま引き継げない．昨年度については，”samurai”言語というオリジナル言語で Web 上で動作していた．今年度は，”Gunbai Script 2013”言語というオリジナル言語で実行スクリプトにより c++に変換後，g++で実行オブジェクトを生成し，そのオブジェクトを実行しログが出力する．そのログを Java で作成されたビジュアライザでゲーム画面表示できる.

2.2　ルール (Rule)

2.2.1　ゲーム進行

チーム数

予選：国内：24 チーム，　海外：4 チーム
決勝：国内：10 チーム，　海外：2 チーム

ラウンド数

各チーム 20 ラウンド

１ラウンドのターン数

最大 300 ターン

エージェントの行動

ターン毎に全てのエージェントが 0 ～ 5 までの行き先に移動
また行動しないが選択可能

2.2.2 ラウンド開始 (初期状態)

1 ラウンドのプレイ人数

赤，緑，青，黄の 4 チーム

1 チームのエージェント数

侍エージェント：3 人
犬エージェント：1 匹

盤面 (Board)

正六角形のタイルが敷き詰められる．
行： ２５～３５行から毎ラウンド設定
列： ２５～３５列から毎ラウンド設定

門 (Gate)

上下左右対称に 2 個づつランダムにゲートが配置

エージェントの初期配置

エージェントに不平等が生じないように，上下左右対称にランダムに配置

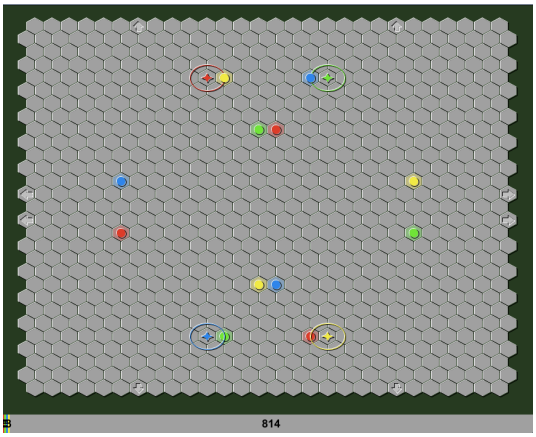

図 2.2.1 スクリーンショット (初期状態)

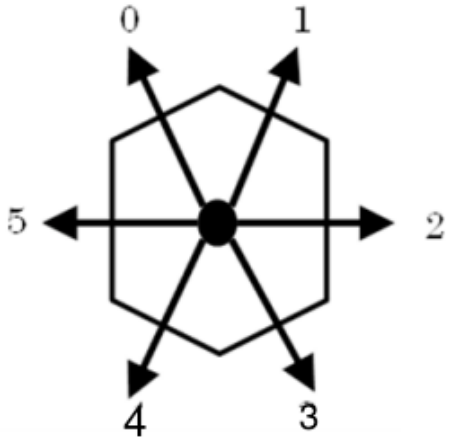

Figure 2 Move Directionss

図 スクリーンショット (進行方向)

侍エージェントの移動 (役割)

足跡を増やす．(既に足跡がある場合は上書き)

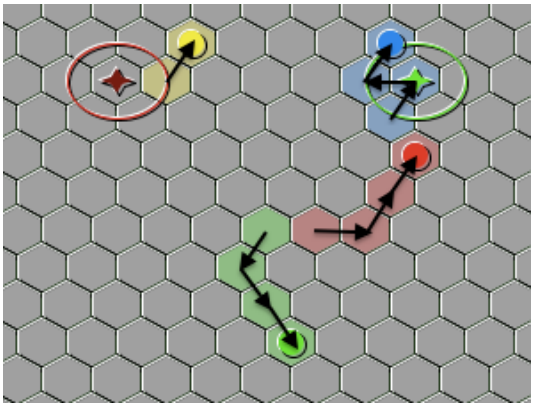

図 2.2.2(1) スクリーンショット (侍エージェントの移動)

犬エージェントの移動 (役割)

他のエージェントの動きを妨害する．
下の図 2.2.2(2) は黄色の矢印は移動可能だが，黒い点線の矢印は移動出来ない．

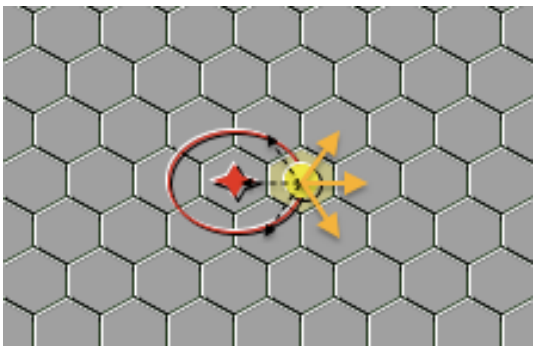

図 2.2.2(2) スクリーンショット (犬エージェントの役割)

フリーズ状態

エージェントが特定の位置に動こうとしたときに発生．発生した後のエージェントは 1 ターン動けなくなる．フリーズしたあとに -1 をリターンすると再度行動が可能となる．

- 他の Agent との衝突
  他のエージェントがいた位置に行こうとするとフリーズ
  他のエージェントと移動先が同じ場合お互いにフリーズ
- 壁に衝突
- 敵の犬エージェントの周囲 1 マスに移動しようとするとフリーズ．ただし，侍エージェント，犬エージェントの順番で動く

大直列交換

1 匹以上の犬エージェントと 3 人以上の侍エージェントが一直線上に並んだとき大直列交換が行われる．一番外側同士，内側同士の位置がスワップされる．

図 2.2.2(3)　スクリーンショット (大直列交換前)

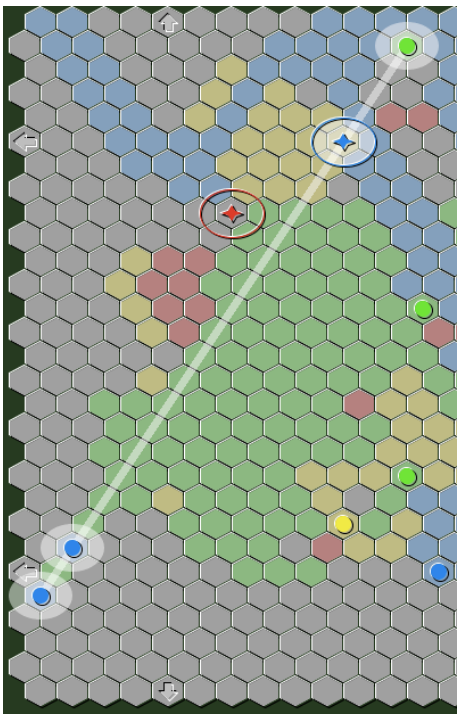

図 2.2.2(4)　スクリーンショット (大直列交換後)

包囲

領土を自分の色で囲った場合，囲った陣地がすべて自分の色になる．またその中にいるエージェントはポイントを増やせない．

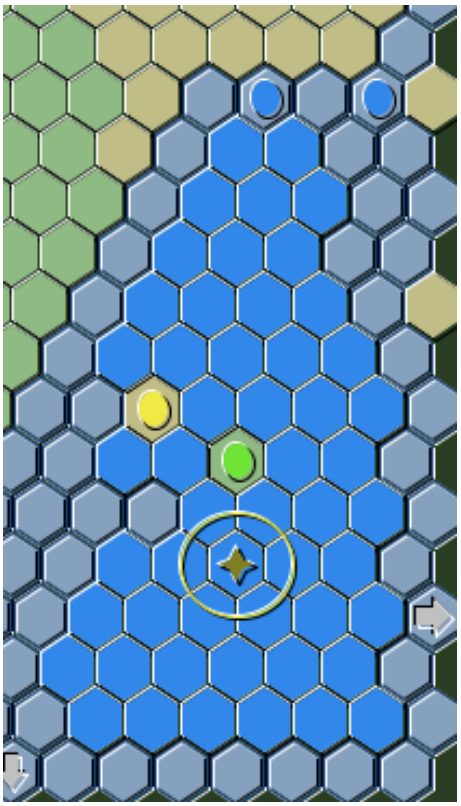

図 2.2.2(4)　スクリーンショット (包囲)

2.2.3　ラウンド終了

- 大陸横断
  盤面の上辺下辺，または右辺左辺が自分のチームの色で繋がることを大陸横断という．大陸横断をしてさらに他のチームよりも占領地が多い場合，即座に試合が終了する．他のチームは占領地が多い順番に順位が決定する．

- 大陸占領
  ゲーム終了時に，占領地が多い順に順位が決定する．

2.2.4　コスト

プログラム処理にはコストが設けられている．条件式，繰り返し文，変数，配列，定数，演算子を使うごとに 1 づつコストがたまり，関数を作ると使うたびに中身の 2 倍コストがかかってしまう．合計１００万コストが上限になっており，それを超えた場合プログラムは動かない．

2.2.5　前年度と今年度のルールの違い

盤面の大きさが固定からランダムに．エージェントの初期位置がランダムに．犬エージェントは侍エージェントの進行の妨害のみ．前年度のルール捕縛（隣接している相手をずっと動けない状態にする）はできない．前年度は大陸横断したら即座に試合終了だったが，今年度は大陸横断してポイントが一番高い場合即座に試合終了に変更された．

3　SamurAI Coding：戦略

3.1　侍エージェントの動き方の方針

昨年のルールから侍エージェントの動きの方針は基本的には２通りある．１つ目は，序盤で一気に大陸横断を狙う方法．２つ目は，相手の邪魔をしつつ自分の領土を広げ大陸占領を狙う方法である．去年は大陸横断をしたチームが即座に勝利できたため、大陸横断優先のチームが多かった．今年は大陸横断をしたチームのポイントが一位にならなければいけないため，今年は大陸占領を狙う方針を採用した．これは大陸占領をすることで大陸横断優先のチームよりもポイントが多く稼げるため，ラウンド毎の順位が高くなりやすいからである．

3.2　フリーズ状態をなくす関数を作成

去年の先輩のプログラムでは侍エージェントと侍エージェントが重なるときにフリーズ状態で動かない状態になってしまうことがあった．

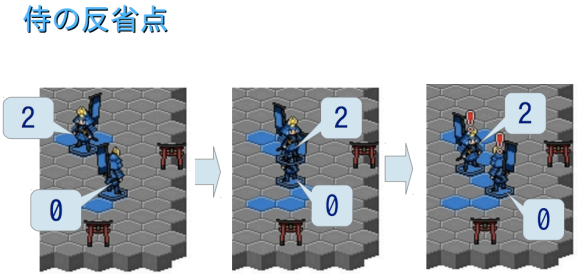

図 3.2　スクリーンショット (昨年度侍エージェントの反省点)

今年の目標は，まずフリーズ状態が起こらないように行き先を決定する priorityMove 関数を作成した．まずフリーズ状態について説明する．フリーズ状態になるときはルールの所で説明した通り壁との衝突，他のエージェントとの衝突，敵チームの犬エージェントの周囲に移動しようとしたときにフリーズ状態が発生する．フリーズ状態になると次のターンは必ず動けなくなりそのエージェントの配列に-1 を返さない限り動くことはできない．そのため，この状態になったときに 1 ターン無駄になってしまう．それを避けるために priorityMove 関数が必要となる．priorityMove 関数はエージェントの番号と優先的に移動したい行き先 0 から 5 までを引数に入れると，ほぼ確実に移動できる位置を返す関数である．

以下，priorityMove 関数のアルゴリズムである．

priorityMove 関数 アルゴリズム

1. 行き先のポイントを付ける ret[6] という配列を用意する．
2. 初期化 ret[6] の全ての要素に 1 を入れる．
3. 行き先が壁ならば ret[i] = 0
4. 壁でない所を見て自分の色ならば ret[i] = 3
5. 行き先に他のエージェントが存在するならば ret[i] = 0
6. 行き先のタイルが他のエージェントと隣接している場合 ret[i] = ret[i] * 2
7. 行き先のタイルが他のチームの犬エージェントと隣接している場合 ret[i] = 0
8. ret 配列の要素を k = 1..7 まで探索し，最小の値の所を優先的に返す．

3.3 侍エージェントの動き

六角形を描き，包囲して大量にポイントを獲得する．包囲の成功は他チームのエージェントの動きによって左右されやすいが，成功した場合多くの領土を占領でき，上位になりやすくなる．

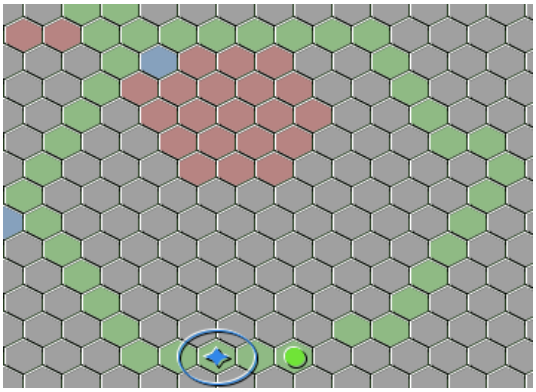

図 3.3(1) スクリーンショット (六角形を描く)

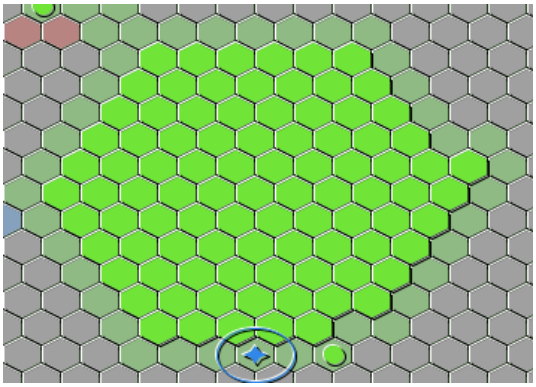

図 3.3(2) スクリーンショット (六角形を描く)

一人の侍エージェントは端についたら周りを動き大陸横断を狙う．

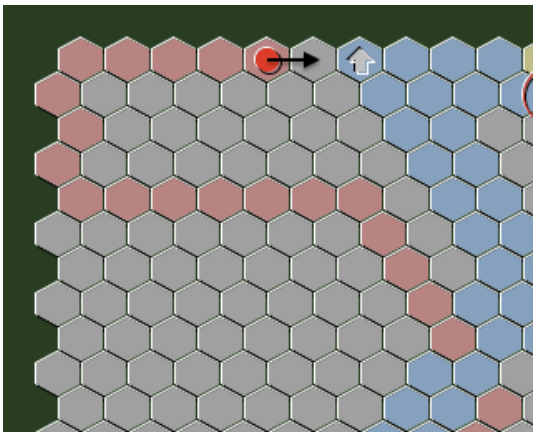

図 3.3(3) スクリーンショット (壁の周りを動く)

3.4 犬エージェントの動き

最短距離にいる敵エージェントを追跡する．

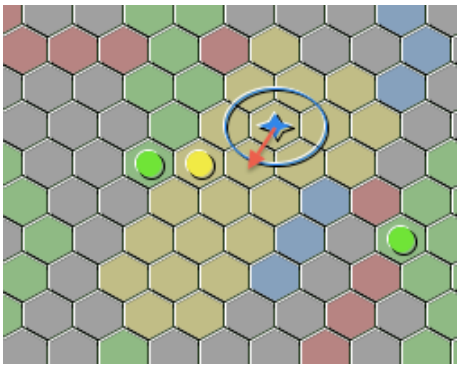

図 3.3(4) スクリーンショット (犬エージェントの動き)

4 SamurAI Coding：予選の結果，考察

２４チーム中７位になり予選突破した．予選を通過出来るのは国内上位１０チーム，海外上位２チームである．フリーズ状態にならない関数，六角形を描くという作戦はある程度有用であった．４位以下のチームでは一番ポイントが高いため，大陸横断はあまり出来なかったが平均順位が高かったと推測出来る．

Japan-Domestic Preliminary Tournament

	Team	Points	Ratio Sum	Area Sum	Qualified
1	SC'13	87	6138128	4175	yes
2	Chime a Nimble Grill	72	5716027	3849	yes
3	ThreeStones	71	5332766	3750	yes
4	Bream	70	5253431	3052	yes
5	SUIHANKI	69	5206698	3181	yes
6	nishimo	68	4744585	2721	yes
7	Algori	66	5339759	3383	yes
8	kobaski	65	5145973	3075	yes
9	arakawaken	64	4955455	3283	yes
10	OHNISHI LAB IN THE SKY	63	4956722	3155	yes

図 4.1(1) スクリーンショット (予選の結果)

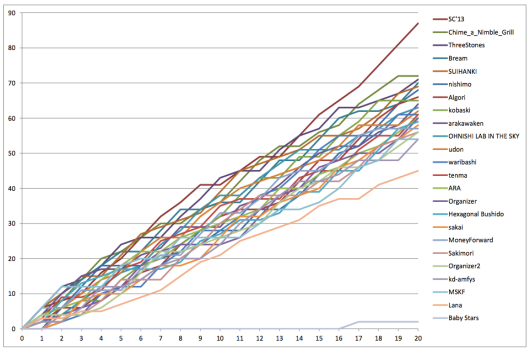

図 4.1(2) スクリーンショット (予選のグラフ)

4.1 決勝へのプログラム改良

予選で提出したプログラムではある条件になるとフリーズが発生してポイントを稼げなかった．それを直しさらにフリーズ状態にならないプログラムにする．

予選のプログラムでは行き先がゲートの場合，ゲートの向こうにエージェントが存在した場合フリーズ状態が発生していた．ゲートの向こう側にエージェントが存在するかどうかを識別するプログラムを追加し，エージェントがいた場合は進行を変えるようにした．

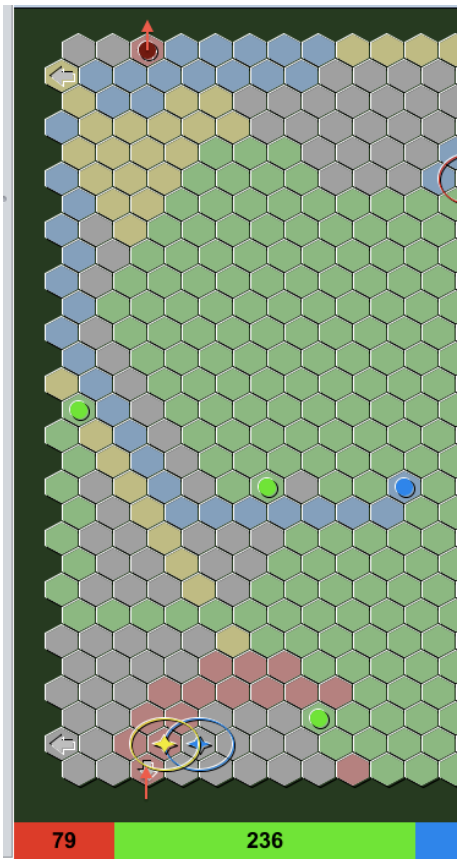

図 4.1(1) スクリーンショット (ゲートの先を見る)

ある条件で味方の侍エージェントが同じ座標に進もうとするためフリーズ状態が発生していた．これを防ぐため，一度フリーズ状態になった行き先を再度選択しないようにした．

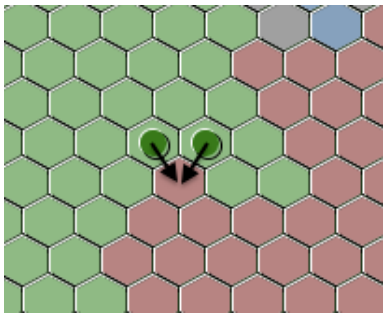

図 4.1(2) スクリーンショット (同進行方向の変更)

5 SamurAI Coding：決勝の結果，考察

決勝の結果は１０位中６位となり一応世界６位という結果を収めた．他のチームとの対戦を見るのは決勝だけなので，このとき初めて様々な戦略を見る事ができた．

	Team	Points	Ratio Sum	Area Sum	Script
1	SC'13	88	5777916	3872	script
2	Bream	71	5186305	3415	script
3	SUIHANKI	68	5268622	3737	script
4	Chime_a_Nimble_Grill	66	5509714	3734	
5	nishimo	63	4750154	2721	
6	Algori	62	5244577	3460	
7	kobaski	62	5140585	3481	
8	OHNISHI LAB IN THE SKY	60	4486366	2789	
9	arakawaken	58	4976178	2829	
10	ThreeStones	56	5011719	3420	
11	HKT	34	4327609	2318	
12	Koala	32	4320140	2274	

図 5(1) スクリーンショット (決勝の結果)

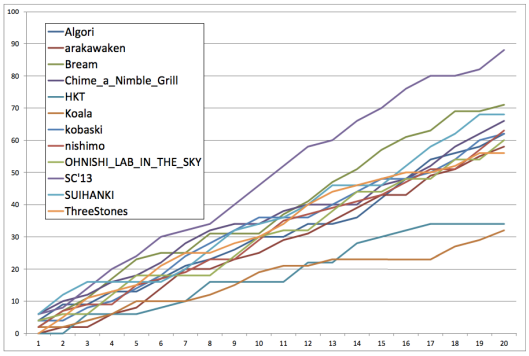

図 5(2) スクリーンショット (決勝のグラフ)

5.1 3 位のチームのプログラム解析

3 位のチームのプログラムはバグで動かないことが多かったため，それが無ければ 1 位と互角に渡り合えるプログラムになったであると対戦を見て思った．主に近い距離にある自分の色に評価（ポイント）を付けてそこを繋げて包囲を狙う．評価する所が無い場合はランダムな動きをする．

5.2 2 位のチームのプログラム解析

2 位のチームはまず小さな円を描きポイントを稼いで大陸横断を妨害する．そのあと，3 人の侍エージェントが画面より右側を限定してポイントを稼ぎ大陸横断を目指す．画面右側にエージェントがあまり存在しない場合，大量に包囲してポイントを稼ぎ，大陸横断をしていた．

5.3 1 位のチームのプログラム解析

自分のポイントがトップのときに最短で大陸横断を狙う．渦巻き型でポイントを増やし敵エージェントが妨害しようとしたときに包囲してポイントを増やす．そのあと大陸横断を狙う．1 位のチームはラウンドが進み強いチームと対戦するときには他のチームに邪魔されてポイントを稼げていなかったが，大陸横断をするスピードは一番最速だった．

5.4 今回の反省点

自分のチームのポイントが一番多くなったら大陸横断を目指すプログラムに変更すべきであった．犬エージェントはもう少し敵エージェントを妨害するプログラムにするべきだった．評価関数を使ったプログラムを作成したかった．

6 終わりに

稿では SamurAI Coding におけるルールから戦略などを細かく説明してきたが，より強い AI を作成するにはヒューリスティック関数（評価関数）を使えばより AI としての知識が深まるのではないだろうか．また予選を通過すると 1 年間情報処理学会の無料で会員になれること，またエクスカーションが開かれグリー株式会社や株式会社 Preferred Infrastructure などに企業訪問が出来る．本年は MarioAI の大会は開催されたか分からないが，プログラムを試作した．ゲーム AI に興味がある人は是非やっておくべきだ．

参考文献

[1] 著者：太原 育夫 , タイトル：新人工知能の基礎知識

[2] SamurAI Coding 2013. http://samuraicoding.info/, (参照 2014-02-10)

[3] marioAI google code. https://code.google.com/p/marioai/, (参照 2014-02-10)

[4] 情報処理学会 http://www.ipsj.or.jp/index.html, (参照 2014-02-10)

[5] 人工知能学会 http://www.ai-gakkai.or.jp/ (参照 2014-02-10)