Vol.2013-CE-118 No.19
2013/2/9

ボードゲームの戦略プログラミングを題材とした Java 演習支援 －ローカル支援ツールの統合とサンプル素材の利用法－

山田航平[†1]　　富永浩之[†1]

問題解決型の応用プログラミングとして，競争型学習を取り入れ，ボードゲーム戦略を題材とする対戦形式での Java 演習を提案している．ローカル側とサーバ側の支援環境 WinG を開発し，2005 年度から授業実践を行っている．ローカル側 LA として，戦略のテスト実行を効率的に行う 4 つのモジュール群を実装した．これらを 1 つの支援ツールに統合した．また，デバッグの参考となる戦譜や局面などのサンプル素材を用意する．これらにより，演習の進行の各段階において，戦略プログラムの作成と改良を効果的に支援する．演習の進行を 4 段階に分け，それぞれでの WinG-LA の利用法を述べる．

Programming Exercise for Problem Solving with Board-Game Strategy - Integration of Local Support Tools and Sample Materials -

KOHEI YAMADA[†1]　HIROYUKI TOMINAGA[†1]

We have proposed an applied Java programming exercise by board-game strategy for problem solving learning. During implementation of hand method of Gogo game, students learn realization of idea as algorithm and revision with trial and error by execution result. We also have developed support system WinG, which consists of the local review package LA and the contest support server CS. The package consists of four modules for execution tests and debugging of strategy under construction. The server maintains a preliminary and the final league, which decide students' score by the result of round-robin matching. We have performed an educational practice since 2005. We integrated four modules of WinG-LA It works in collaboration with some sample data. We consider four phases in progress of exercise. We describe the usage of WinG-LA according to the phases.

1. はじめに

大学情報系では，C 言語などの入門的なプログラミング演習が必修とされる．続いて，応用的な演習では，C++/Java 言語などオブジェクト指向の導入，ソフトウェア開発手法の実践などが中心となる．また，データベースやネットワークなど，多様な情報系技術との融合も扱われる．さらに，特定分野の課題に対する問題解決の手段としてのプログラミングが重視される．

このような総合的な能力の向上を教育目標とするには，指示された通りの要件や仕様を満たすプログラムを作成するだけでは不十分である．むしろ，オープンプロブレムの課題を提示し，各自の創意工夫を取り入れる余地がなければならない．また，コンパイルに成功し，サンプルデータで動作すれば終わりではなく，試行錯誤しながら改良と修正を重ね，効率性や精度など品質の向上を図る過程こそが重要である．

そのためには，魅力的な題材と効果的な演習方法が求められる．しかし，学生の興味と程遠い題材では，具体的なイメージが湧かず，プログラミングの到達目標を描きにくい．また，成果物としてのソフトウェアへの愛着が得られない．そこで，知識情報処理の分野では，ゲーム戦略を題材とする演習が試みられている．これには，競争型学習の要素も盛り込まれ，動機付けへの効果が期待される．

2. 本演習の概要

2.1 ボードゲームと戦略プログラミング

以上の背景に基づき，本研究では，ボードゲーム戦略を題材とする対戦形式での Java 演習を提案している[1][2]．ボードゲームとしては，五目並べに石取りを加えた，二抜き連珠のルールを整備し，五五と名付ける(表 1)(図 1)．五五ゲームは，石を取ることで局面が大きく変化する．連と取という 2 つの勝利条件があり(表 2)，それぞれに攻撃と防御の優先度が考えられ，初心者の段階でも戦略の個性が出やすい(図 2)．

問題設定として，Java 言語で作成したゲーム実行ライブラリを提示し，13×13 の盤面での五五の戦略を Java プログラミングで実装させる．実行ライブラリのオブジェクト構成は，図 3 の通りである．学生は，Computer クラスを継承したサブクラスで，着手メソッド calc_hand()をオーバーライドする．calc_hand()は，局面を引数とし，次の 1 手を返す．局面は，State クラスのインスタンスで，盤面の石の配置や取った石の個数を保持している．

対戦では，先手後手の 1 組で 1 試合とし，勝敗で勝ち点を付ける．1 勝 1 敗では，取った石の数で優勢を決め，同

†1 香川大学
Kagawa University

数は引分とする．戦略の評価は，総当り対戦での勝ち点の合計で順位を決める．ただし，全体の評価は，戦績だけでなく，戦略の自己評価を行った総括レポートも加味する．

2.2 戦略の作成手順と支援環境 WinG

戦略の作成手順は，図 4 のように，戦略方針に従って，各枡の評価値を求め，最高点の位置を着手とする．評価値は，経験的に割り当てた値から，実戦を通して調整していく必要がある．また，局面パターンのより詳細な判別に基づいて精密化していく．学生には，プロトタイプのソースコードを提示し，最低限必要な処理をコメントで指示しておく．典型的な配置パターンの実装から始め，独自の局面分析に進んでいく．

このような戦略作成を支援するため，支援環境 WinG を開発している(図 5)．ローカル側 WinG-LA では，戦略のテスト実行を効率的に行うモジュールを提供し，戦略検討に用いる各種のサンプルを用意する．サーバ側 WinG-CS では，提出された戦略同士を対戦させる大会を運営し，ランキングや戦績を公開する．これにより，試行錯誤的なプログラミングを体験させ，持続的な戦略修正への動機付けを行う．

これまで，2005 年度から数年間，知識情報処理の演習として，五五ゲームを採用した授業実践を行ってきた[3][4]．これらの総括として，学生のプログラムとレポートを分析し，教育効果を評価し，演習方法や支援環境にフィードバックさせてきた．本論では，以上を踏まえ，システムのローカル側 WinG-LA の改良と，サンプル素材を用いた利用法について述べる．

2.3 大会運営サーバ WinG-CS

作成中の戦略にフィードバックをかけて，持続的に演習に取り組ませるため，最終大会の締切までを予備戦期間とする(図 6)．予備戦期間中に提出された戦略は，サーバ上で他の戦略と対戦し，定期的に結果が更新され，順位が公開される．順位の推移を見て，自分の戦略を再検討し，状況に応じて戦略を修正していく．予備戦後に，提出した戦略の強さを総合的に判断し，最終大会の戦略を選択する．これらの戦略同士で総当り戦を行う．この結果から最終順位を決定し，成績に反映させる．このように，自分の戦略を常に評価する機会を設けることで，試行錯誤の繰返しを動機付ける[5][6][7]．

以上を支援するため，大会運営サーバ WinG-CS を運用する．システムは，ユーザ管理部，戦略提出部，戦略管理部，全体結果部，予備対戦部，最終対戦部の各モジュールから構成される．Java での対戦実行は JDK1．7，内部処理は Ruby1．9．2 で実装し，DB は XML と SQL を用いる．

戦略提出ページでは，アップロードする戦略に，名前やコメントを付けられる．戦略が実行可能かどうかを学生に通知する．順位表示ページでは，全戦略の総当り戦の結果を勝率順に表示する(図 7)．最強戦略による個人毎の順位も表示する．戦略管理ページでは，自分が提出した戦略について，戦績などを集約して表示する．対戦履歴ページでは，対戦ごとに，勝因や手数などを確認できる．指名対戦ページでは，任意の戦略を選んで対戦し，戦譜を再現できる．ただし，他人のソースコードを閲覧することはできない．最終結果ページでは，各自が選択した戦略で総当り戦を実施し，勝点で順位表示する．

表 1 五五のルール

着手	先手が黒石，後手が白石を使用し，交互に打つ 2 個並んだ相手の石を両側から挟んで取れる 1 手で複数の方向の 2 連を同時に取ることも可能 後から石間に置いて 2 連になったものは取れない
勝利条件	完全な五連を作るか，10 個(5 回)石を取ると勝ち 完全な五連とは，挟んで取られない五連のこと 長連は五と認められない
禁手	「三々」は，先手後手共に禁手である 石を取った後の「三々」は，禁手とならない

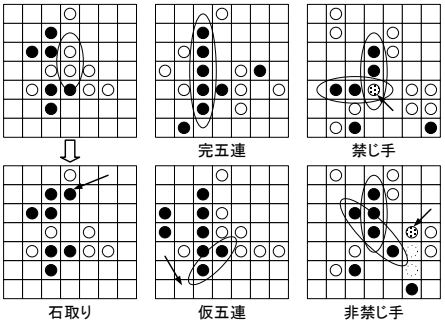

図 1 五五のルールと局面

表 2 勝敗判定の手順

1. 置けない打手(盤外，重置，連打)は反則負け
2. 打って(石取り前に)三々になったら，禁手で負け
3. 石を取っても，相手の五連を崩せなければ負け
4. 相手の長連から石を取り，五連ができたら負け
5. 自分で石を取り，五組に達すれば勝ち
6. 相手に崩されない五連を作ったら勝ち
7. 全ての枡が埋まっていたら引分
8. 時間内に打たないときは投了で負け

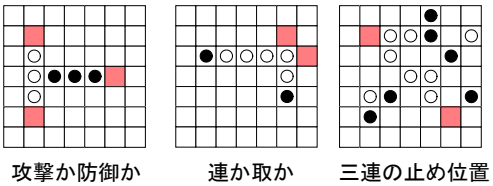

図 2 戦略の分岐点

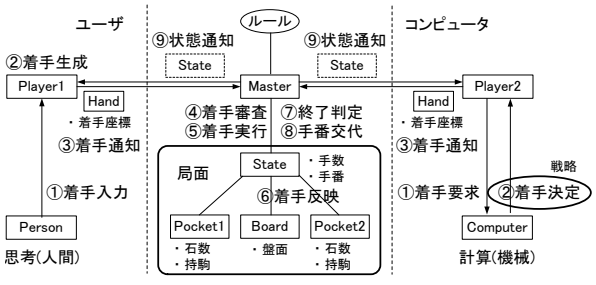

図 3 WinG の実行ライブラリ

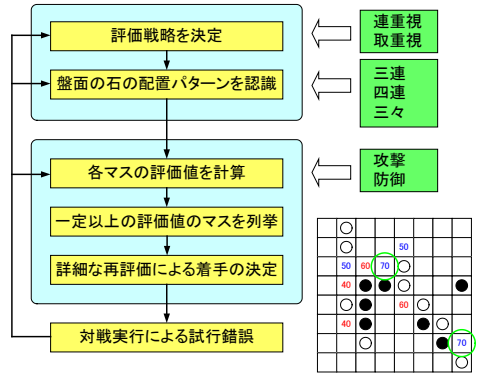

図 4 戦略プログラムの組立て方

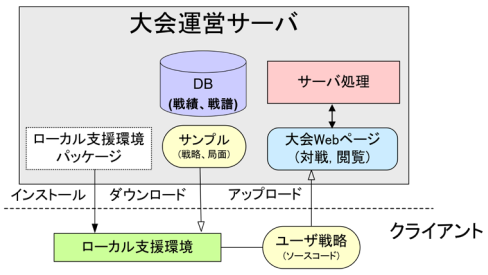

図 5 支援環境 WinG の構成

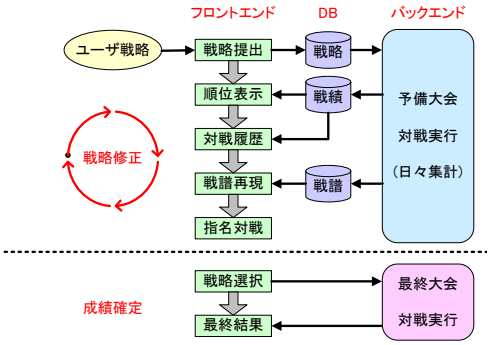

図 6 予備大会と最終大会

■ 順位

ダミープレイヤ無しの順位表

順位	勝率	戦	勝	敗	分	点	戦略名	作成者	登録日時
1	0.625	16	10	4	2	40	Player05_02	Player05	2012.01.14 13:48:40
2	0.579	19	11	6	2	46	Player02_01	Player02	2012.01.14 13:35:34
3	0.562	16	9	6	1	36	Player02_02	Player02	2012.01.14 13:47:39
4	0.562	16	9	7	0	37	Player03_02	Player03	2012.01.14 13:47:56
5	0.562	16	9	6	1	38	Player01_04	Player01	2012.01.14 14:04:07
6	0.526	19	10	9	0	40	Player01_01	Player01	2012.01.14 13:34:30
7	0.500	16	8	7	1	31	Player04_02	Player04	2012.01.14 13:48:16
8	0.500	16	8	5	3	36	Player01_03	Player01	2012.01.14 14:03:54
9	0.438	16	7	8	1	29	Player01_02	Player01	2012.01.14 13:47:17
10	0.421	19	8	8	3	36	Player04_01	Player04	2012.01.14 13:36:12
11	0.421	19	8	11	0	34	Player05_01	Player05	2012.01.14 13:36:32
12	0.417	12	5	7	0	21	Dummy_02	Dummy	2012.01.14 14:07:51
13	0.417	12	5	6	1	22	Dummy_04	Dummy	2012.01.14 14:11:24
14	0.333	12	4	7	1	20	Dummy_05	Dummy	2012.01.14 14:11:35
15	0.333	12	4	7	1	22	Dummy_03	Dummy	2012.01.14 14:08:01
16	0.263	19	5	10	4	31	Player03_01	Player03	2012.01.14 13:35:54
17	0.250	12	3	7	2	18	Dummy_01	Dummy	2012.01.14 14:07:35

図 7 大会運営サーバ WinG-CS の動作画面

3. ローカル支援ツール WinG-LA

3.1 WinG-LA の全体構成

ローカル支援ツール WinG-LA は, 学生の躓きを減らし, 全体的な戦略のレベルアップを図るための支援を行う. WinG-LA およびゲームの実行ライブラリは, 大会運営サーバから事前にダウンロードしておき, 戦略検討の各種のサンプルと合わせて利用する. プログラミング自体は, 既存のテキストエディタや, Eclipse などのプログラミング統合環境で行う.

WinG-LA は, 4 つのモジュールから構成される(図 8). 対戦実行モジュールは, 実装した戦略の確認のため, 人間または他の戦略プログラムとの対戦を行う. 局面生成モジュールは, 着手のデバッグのため, 任意の初期局面を生成する. 戦譜再現モジュールは, 参考とすべき戦譜からの対戦を鑑賞する. 着手確認モジュールは, 指定された戦略と局面に対して, 次の一手のみ実行する[8][9].

今回, 従来は別個に起動していた各モジュールを統合し, 1 つの支援ツールへと改良した. 各モジュールは, タブ形式の GUI で, 自由に切り替えることができる. 各モジュールが扱うデータも共有され, 機能の有機的な連携を実現する. ただし, 現時点では, 対戦実行モジュールで用いる戦略プログラムは, WinG-LA 本体を起動する前に, 予めコンパイルして, 組み込んでおく必要がある. この点については, 戦略プログラムを動的に追加できるように, 改良する予定である.

3.2 対戦実行モジュール

対戦実行モジュールは, 試作した戦略プログラムの実行テストに用いる. 試作戦略と, 人間または予め用意されたサンプル戦略を対戦させ, 戦略の強さや動きを確認する. また, 実装した戦略が仕様通りに動かないという学生へのテスト支援として, 対戦実行時に, 与えられたサンプル局面を初期局面として指定できる. 初期局面は, ゲーム開始時の石の配置と取石数を指定するデータである. テストしたい特定の初期局面からスタートできるので, テストやデバッグの効率が上がる. 対戦の過程は, 戦譜として保存でき, 戦譜再現モジュールで, 後から見直せる.

GUI は, 図 9(a)の通りで, 局面読込, 対戦指定, 対戦実行, 結果保存の各機能を提供する. 局面読込では, 対戦の初期局面としたい局面データを読み込む. 対戦指定では, 既にコンパイルされている戦略の中から, 対戦相手を指定する. 対戦実行では, 人間が対戦する場合は, 自分の手番で, 盤面上の位置をマウスでクリックして指定する. 盤面の左右には, 両者の取石数が表示される. 下側には, それまでの戦譜が表示される. コンピュータ同士が対戦する場合は, ほぼ一瞬で対戦が終了する. この場合, 戦譜再現モジュールで, 対戦の過程を確認する.

対戦の終了時には, 勝敗と勝因が表示される. 勝因は,

五取，五連，禁手，反則に分かれる．人間の場合は，反則に当たる範囲外や重複手は無視されるが，コンピュータの場合は，敗北となる．また，一手にかかる時間が制限を超える場合も，反則による敗北となる．先手後手が交替した 2 試合を 1 対戦とするので，1 試合目が終了すると，2 試合目に突入する．2 試合目が終了すると，対戦としての結果(完勝，僅勝，引分)を表示する．僅勝とは，1 勝 1 敗で取石数による優勢勝ちを指す．戦譜保存では，戦譜データに名前を付けて，指定した場所に保存する．

3.3 戦譜再現モジュール

戦譜再現モジュールは，与えられた戦譜から対戦を再現する．戦譜再現は，対戦実行モジュールでの実行結果を分析し，戦略を修正するのに用いる．サンプル戦譜をいくつか配布し，なかなか良い戦略にならないという学生に，戦略を推定させ，参考になるように支援する．また，再現途中の局面を初期局面として保存することで，擬似的に対戦を途中からやり直すこともできる．悪手を打った戦略を修正した後，同じ局面で違う着手ができているかを確認するという使い方が考えられる．

GUI は，図 9(b)の通りで，戦譜読込，自動実行，手動実行の機能がある．戦譜読込では，予め用意されたサンプル戦譜や，対戦実行モジュールで生成されたユーザ戦譜を呼び出す．戦譜は，先手後手が交替した 2 試合を 1 対戦とするので，どちらの試合かを指定する．自動実行では，1 手ごとの経過時間を指定し，開始ボタンで戦譜の再現が始まる．右側に表示される戦譜から，任意の局面を指定して，途中から始めることもできる．手動実行では，ボタンのクリックで 1 手ずつ進める．1 手を戻すこともできる．

3.4 局面生成モジュール

局面生成モジュールでは，学生が既存のサンプル局面を修正したり，自分で初期局面を作成する．あるパターンに該当する初期局面からの対戦により，評価値の仕様通りに着手が行われているかの確認が容易になる．例えば，必勝局面における最適な攻撃の手(五連，五組など)を着手しているかを検証する．

GUI は，図 9(c)の通りで，局面読込，盤面編集，局面設定，局面整理，局面保存の各機能を提供する．局面読込では，空の局面から始めるか，既存の局面を呼び出す．盤面編集では，黒または白の石を選んで，盤面に自由に配置する．既に置いてある石を削除することもできる．これによって，試してみたい初期局面を生成する．黒と白を自由に配置できるので，通常のゲーム進行からは得られないような局面も可能である．局面設定では，盤面上の配置だけでなく，取石数も設定する．先手後手のどちらの局面かを指定する．先手後手の入替えのためには，黒と白を一括して反転する．

局面整理では，局面に名前を付ける．その際，複数の局面をパターンごとに整理するため，パターンのラベル名と個々の局面の名前に分ける．既存のパターンに関連付けたり，新たなパターンとして新規のラベル名を用意することもできる．局面保存では，局面を保存する．保存先は，パターンごとに指定する．

3.5 着手確認モジュール

着手確認モジュールは，上記の 3 つのモジュールを補完するため，2011 年度から追加した．以前は，ある局面に対する戦術のみを検証する機能が不十分であった．例えば，ユーザが，五連崩しの戦略のアルゴリズムを設計し，実装したとする．作成したプログラムが適切に動作しているか確認するため，対戦実行モジュールを用いてテストを行う．五連崩しの石の配置を含む局面データを初期局面として読み込み，対戦を実行し，着手を確認する．その結果，ある局面では適切に動作し，別の局面では不適切に動作することが分かったとする．そこで，ユーザは，様々な局面パターンで調査し，原因を究明しようとする．しかし，対戦実行モジュールでは，局面ごとに読込みが必要で，かつ，対戦は勝敗が決するまで続くため，作業が煩わしく，テスト効率が悪い．

着手確認モジュールは，より効率的な検証を支援するため，次の要件に基づく．指定された戦略と局面に対して，次の一手のみ実行する．実行結果について，適切な着手か判定する．また，テストを効率よく行えるよう，複数の局面に対して一括して実行する．個々の結果を総合的に評価する．機能は，複数の初期局面を一括して読み込む．各局面に対して，良手、悪手となる枡を指定する．試行する戦略を選択する．各局面に対して，次の一手のみを連続して実行する．指定された良手、悪手と実際の着手を照合して結果を表示する．

GUI は，図 9(d)の通りで，局面選択，着手指定，戦略選択，着手実行，結果表示の各機能を提供する．局面選択フェーズでは，石の配置パターン毎に分類された初期局面データを選択する．着手指定フェーズでは，ユーザが改良した戦略が置くことを期待する/しない枡を，盤面に設定する．着手実行フェーズでは，各局面に対して，実際にどのような一手を打つか試行する．結果表示フェーズでは，個々の照合結果と正答率を表示する．

3.6 サンプル素材

サンプル素材は，戦略，戦譜，局面の 3 つを提供する．サンプル戦略は，ソースコードを公開しているものと，バイナリのみのものがある．前者は，プロトタイプ戦略を指し，基本的な戦略をコメントで示している．また，基本的な石の配置パターンを認識するライブラリを持つ．例えば，ランダムな着手を行うもの，不十分なパターン判定のままのもの，防御に徹するものなどを用意する．そのソースコードを参考に各自のプログラムを修正させる．コードを再利用することで，学生を早い段階から高度な戦略の実装に取り組ませる．後者は，適当な強さの戦略を用意する．弱

い戦略は，練習相手として，乗り越えるべき最低基準を示す．強い戦略は，過去の実践で上位に入った戦略から選抜したものである．攻撃・防御優先，連・取優先，戦略の強弱など，多彩なプログラムを用意する．プログラムの振舞から戦略を推測し，リバースエンジニアリングさせる．

サンプル局面は，基本的な石の配置パターン(五連崩しや禁じ手の回避など)ごとに複数の局面データを提供する．特定の局面におけるユーザ戦略やサンプル戦略の振舞をテストする際に，初期局面データとして読み込み，利用する．また，着手確認モジュール用として，良手と悪手の位置も含まれたサンプル局面も用意する．

サンプル戦譜は，WinG-LA と一緒に配布されているものと，WinG-CS の予備大会における対戦結果として公開されるものがある．

4. システムの利用手順

4.1 演習の段階における WinG-LA の利用

WinG-LA を用いた演習では，学生の演習の進行段階を大きく 4 つのフェーズに分けて捉える．第 1 フェーズは，プログラミング以前の段階で，領域分析として問題の要件を把握するレベルである．さらに，1-1 他人やコンピュータと対戦してルールを理解する段階，1-2 戦略を検討する段階，1-3 プロトタイプ戦略のソースを理解する段階に細分する．第 2 フェーズは，練習問題として与えられた仕様を満たすプログラミングを行うレベルである．局面パターンの認識を実装する難度で，4 段階に細分する．2-1 初歩的な局面は，ゲーム進行に最低限必要なパターン，2-2 基本的な局面は，飛び形や初歩の亜種，2-3 応用的な局は，パターンの組合せ，2-4 発展的な局面は，大局的な評価が必要な局面の分析を扱う．

第 3 フェーズは，自由プログラミングの段階で，戦略のアイデアを自分で仕様にまとめて実装するレベルである．さらに，3-1 自分で定義した局面パターン，3-2 弱いダミー戦略と試戦する段階に細分する．第 4 フェーズは，改良プログラミングの段階で，非機能要件として最適化を目指すレベルである．さらに，4-1 戦略の優劣を検討する段階，4-2 他の戦略と対決する段階，4-3 順位を上げていく段階に細分する．以下では，このフェーズに沿って，WinG-LA の各モジュールの機能やサンプル素材の効果的な利用法について述べる．

4.2 プログラミング以前の段階

1-1 フェーズでは，最初の取掛りとして，対戦実行モジュールを用いて，サンプル戦略と対戦を行い，ゲーム体験を理解する．雰囲気が掴めたら，学生同士で対戦を行い，ルールを把握する．また，ルールで不明な点があれば，局面生成モジュールを用いて，学生同士でのその局面を作って確認する．1-2 フェーズでは，別途のマニュアルを読み，評価値やパターン認識など，戦略作成の手法を理解する．慣れてきたら，対戦実行モジュールを用いて，色々なサンプル戦略と対戦することで，戦略のアイデアや設計を考える．ユーザ同士の対戦では，ハンディを付けた局面を生成してからゲームを開始したり，注目すべき局面を初期状態のユーザ局面として保存する．戦譜再現モジュールを用いて，サンプル戦譜やユーザ戦譜を読み込み，対戦を再現し，戦略の推定や対戦結果の見直しをして，戦略作成の参考にさせる．再現の途中で，注目すべき局面を初期状態として保存する．1-3 フェーズでは，別途のマニュアルを読み，プロトタイプ戦略のクラス構成を理解する．着手メソッドのソースを開き，プログラムとコメントを読む．

4.3 練習プログラミングの段階

第 2 フェーズでは，スケルトンコードの一部を実装したユーザ戦略をユーザと対戦させ，戦略が反映されたことを確認する．さらに，プロトタイプ戦略を実装し，元のプロトタイプ戦略と対戦してみる．また，着手確認モジュールを用いて，実装したパターン認識に該当するテストケースを選択し，着手を確認する．デバッグのために，自分で作成したユーザ局面も利用する．細分化された各フェーズにおいては，難度に応じた適切なサンプル局面を用いる．

4.4 自由プログラミングの段階

3-1 フェーズでは，戦譜再現モジュールを用いて，ユーザ戦譜を読み込み，対戦を再現し，対戦結果を見直し，戦略作成の参考にさせる．また，局面生成モジュールを用いて，自分で定義した局面パターンに該当するユーザ局面を幾つか生成し，それらにラベルを付けて整理する．既に作成したユーザ局面も，読み込み，修正して，別のユーザ局面として保存する．そして，着手確認モジュールを用いて，自分で考えた局面パターンに該当するユーザ局面を与え，正しく着手するか確認する．3-2 フェーズでは，試作したユーザ戦略を，試しに弱いダミー戦略と対戦させ，戦略修正が有効であったかを実感する．

4.5 改良プログラミングの段階

4-1 フェーズでは，戦譜再現モジュールで再現中の対戦から局面生成モジュールに移り，特定の局面を取り出して，該当する局面パターンに分類する．また，類似のユーザ局面に加工する．4-2 フェーズでは，予備大会中に敗北した対戦を再現し，敗因を分析する．分析結果から戦略を修正し，順位を上げていく．

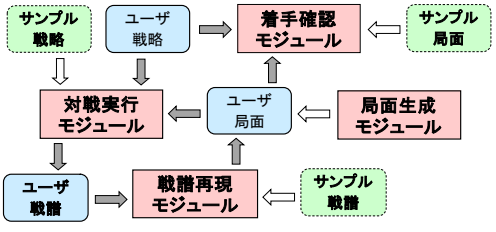

図 8 WinG-LA の全体構成

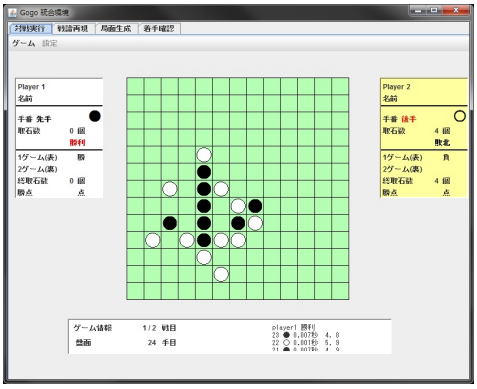

図 9(a) 対戦実行モジュール

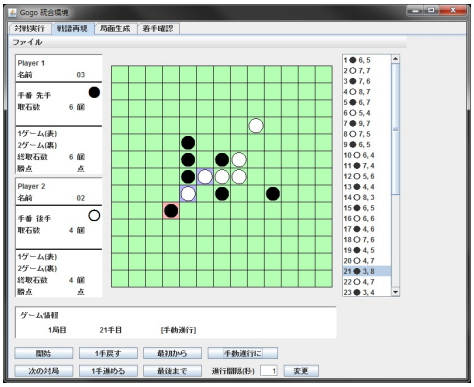

図 9(b) 戦譜再現モジュール

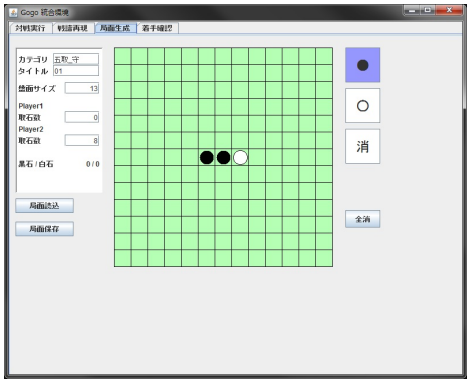

図 9(c) 局面生成モジュール

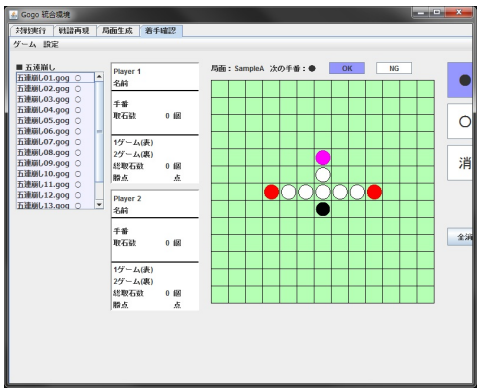

図 9(d) 着手確認モジュール

5. おわりに

これまで提案してきたボードゲーム戦略を題材とする Java 演習において，2012 年度の実践に向けて，支援環境 WinG の機能を強化した．ローカル支援ツール WinG-LA では，着手確認モジュールを含む，4 つのモジュールを，1 つの支援ツールに統合した．各モジュールは，タブ形式の GUI で，自由に切り替えることができる．各モジュールが扱うデータも共有され，機能の有機的な連携を実現する．

また，WinG-LA を用いた演習において，学生の演習の進行段階を大きく 4 つのフェーズに分けて捉えた．このフェーズに沿って，WinG-LA の各モジュールの機能やサンプル素材の効果的な利用法について整理した．

今後の課題として，授業実践で実際に学生に利用させ，機能やユーザビリティに関するアンケートを実施して，ローカル支援ツールの有用性を検証する．また，演習の進行段階における学生の振舞いを捉え，大会運営サーバ WinG-CS の改良や，WinG-LA との連携についても検討し，より高い教育効果を目指す．

参考文献

1) 尾崎浩和，富永浩之：ボードゲームの戦略プログラミングを題材とした Java 演習支援 －五五ゲームの実行環境と大会運営方法－，電子情報通信学会 技術研究報告, Vol.106, No.35, pp.55-60, (2006).
2) 尾崎浩和，富永浩之，林敏浩，垂水浩幸：ボードゲーム戦略を題材とする問題解決型プログラミング演習支援 －試行錯誤的な戦略作成の支援環境とサンプル提示－，教育システム情報学会 研究報告, Vol.22, No.4, pp.69-74, (2007).
3) 山田航平，富永浩之：ボードゲームの戦略プログラミングを題材とした Java 演習支援 －演習実践と対戦結果の分析－，電子情報通信学会 技術研究報告, Vol.111, No.141, pp.59-64 (2011).
4) 山田航平，富永浩之：ボードゲームの戦略プログラミングを題材とした Java 演習支援の実践状況，平成 23 年度電気関係学会四国支部連合大会, pp.312 (2011).
5) 山田航平，富永浩之：ボードゲームの戦略プログラミングを題材とした Java 演習支援 －対戦結果の順位分析と対戦方法の考察－，電子情報通信学会 技術研究報告, Vol.112, No.166, pp.29-34 (2012).
6) 山田航平，富永浩之：ボードゲームの戦略プログラミングを題材とした演習実践の結果分析と対戦方法の考察，教育システム情報学会 第 37 回全国大会, pp.130-131 (2012).
7) 山田航平，富永浩之：ボードゲームの戦略プログラミングを題材とした Java 演習の大会運営サーバの効率化，情報科学技術フォーラム, No.3, pp.627-628 (2012).
8) 山田航平，富永浩之：盤面ゲームの戦略を題材とするプログラミング演習支援システムにおける着手確認モジュール，ゲーム学会 第 10 回全国大会, pp.21-22 (2011).
9) 山田航平，富永浩之：ボードゲームの戦略プログラミングを題材とした Java 演習支援 －着手確認モジュールの導入と大会支援サーバの GUI の改良－，電子情報通信学会 技術研究報告, Vol.111, No.473, pp.19-24 (2011).