第２６回競技部門：30008

チーム名：プログラムが一晩でやってくれました

学校名：八戸工業高等専門学校

1. 開発環境

IDE：Qt Creator

言語：C++

コンパイラ：Visual Studio

OpenMP, Boost

2. GUI

Qt により GUI を作成した。**演算中に解精度の変更**が可能で制限時間に柔軟に対応できる。

メイン（図 1）：問題の送受信、ソルバーの選択、**手動での石の敷詰め**などを行う。

記録（図 2）：問題番号、結果などを記録する。

アニメーション（図 3）：回答結果をアニメーションで表示する。

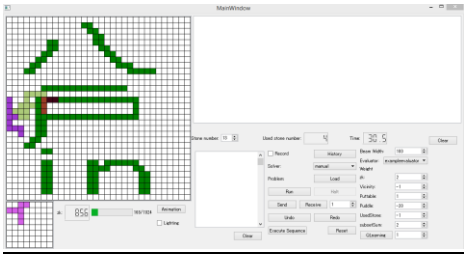

図１　メイン画面

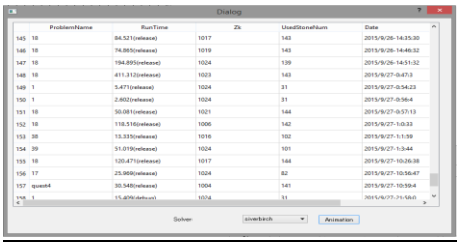

図２　記録画面

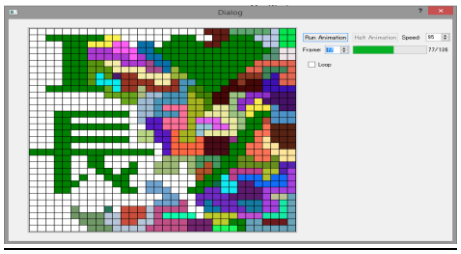

図３　アニメーション画面

3. 解の生成

3.1 アルゴリズム

ビームサーチをベースに開発した。通常のビームサーチはある**一つの評価値**に基づいて並べられた解候補のうちの、上位一定数の解候補のみを保存することで評価値の悪い解候補を削除している。しかし、我々の開発したアルゴリズムではそれぞれ異なる基準で求められる**複数の評価値**を用いた。評価の基準として、石の近傍の数の多さや、石と石の結合度の高さなどを用いた。評価値 1 基準の上位 X 個、評価値 2 基準の上位 Y 個、評価値 3 基準の...というように解候補を保存するようにした。このようにすることで解に多様性が生まれ、様々なケースの問題に対応することができる。また、同じ状態の解候補を複数保存することは無駄であるので、**XOR 演算を応用したハッシュ**により重複を回避した。PC は三台あるので、それぞれの PC で速度特化型、精度特化型、特定の問題に特化した変則型というように異なるアルゴリズムを実行し解を求める。

3.2 高速化

OpenMP により**並列処理**を行う。**石と敷地の情報をビットで表現**することで石を置けるかどうかの判定や、敷地情報の更新を高速な**ビット演算**で行うことができる。

3.3 評価関数

埋まった石の数などの特徴量の重要度に応じて、重みをつけて評価値を計算する。重みは**強化学習**により求められた最適な値とする。また、残り時間や残りの石の数などから**競技の進行度を計算**し、それにより重みの値を変更し**動的な評価関数**となるようにした。