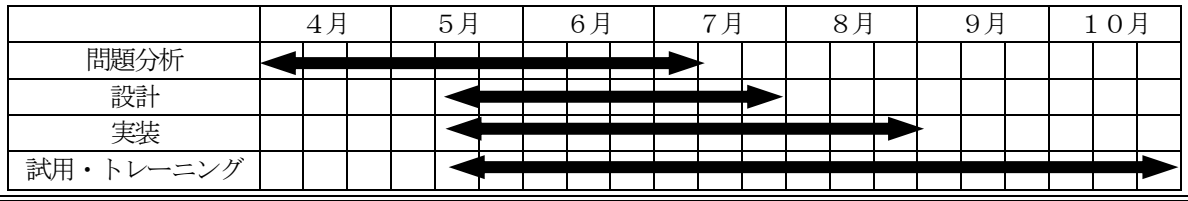
部　　門	競　技　部　門	No.1 登録番号	30019

No.2

1) 予定開発期間：5ヶ月
2) 予定開発人数：3人



No.3

実現方法

1) ボードを最終盤面にするアルゴリズム

以下の手順に沿ってボードを最終盤面にしていく。
なお、行内の並び替えは右寄せ処理を用いず，左寄せだけで行う。

1. 後の操作をしやすくするために最左列は除いて、ゴールの左列から順に、ゴールと一致するピースを現在のボードの右側に揃えていく。
2. 同じ行のピースの左寄せだけでゴールと一致するピースを揃えられる場合は左寄せを行う。
3. ゴールと一致するピースが同じ行にない場合は、他の行から必要な行に持ってくる。このとき、必要なピースは最左列を経由して持ってくることで、行内で既に揃えてあるピースの並びを変えないようにする。
4. 2.3 の手順を繰り返す。
5. 最後に最左列以外が揃ったら、最左列を揃える。

以上の方法により、ボードを最終盤面にすることが可能である。

2) 不一致ピース・手数を少なくする工夫

ここでは上記のアルゴリズムを使用して手数を少なくする方法を挙げていく。

- なるべく大きい抜き型を優先的に使用することで、手数を少なくすることができる。この手法を用いることによって定型抜き型だけでなく、一般抜き型も積極的に使用することができる。
- 最初から動かす必要がないピースについて動かさないようにする。
- 他の行からピースを持ってくる場合は、そのピースが余分にある行から優先して持ってくる。
- ボードサイズがある程度小さい場合には、数回の枝刈りをした深さ優先探索が使用できる。この探索では、より少ない手数で各行に必要なピースを持ってくることを目標とする。その後 1)のアルゴリズムを適用して手数の改善が見られたものがある場合、この手法を採用する。
- 左寄せの段階でビームサーチを用いてより少ない手数を目指す。

ビームサーチ等のアルゴリズムで良い結果を得るためには、なるべく処理を速くすることが重要である。以下にその方法を挙げる。

- 現在の各行の要素をハッシュ化することで、ピースに型抜きを適用することが可能かどうか高速に判別することができる。
- 配列の代わりにビットボードを使用する。
- このアルゴリズムで巨大な抜き型を使用することはボードの端で適用する以外にほぼないため、ある程度使う抜き型の数を絞る。

さらに高速化する手法も今後検討を進めていく。

3) その他（独創的なところ）

OpenSiv3D を用いてビジュアライザを作成し、実行結果の盤面を確認することができる。また、ビジュアライザは手動の操作を取り入れることができ、特に比較的小さい 32x32 の盤面などで有効になる。
制限時間を加味して、ボードのサイズに応じて使用する手法を変える。

No.4

開発環境
使用言語：Python, C++
使用ライブラリ：OpenSiv3D
使用ソフト：Visual Studio Code, Visual Studio