情報学群実験第一最終課題レポート

細川 夏風

2025 年 8 月 4 日

1　概要

今回のカラータイルのプログラムはその列と行についての探索という手法を用いて，実現している．ただ，それにいくかの工夫をすることにより，探索における枝刈りを実現している．それぞれの仕様の実現方法と枝刈りの手法をいかに示す．

2　本文

2.1　仕様 1 の実現方法

ゲーム開始時にランダムな盤面を作りだすのは，まず盤面全体に空白を除いた色をすべてランダムに色を与える．その後，更に固定数の空白数分だけループさせ，Random 関数を使い，その回数ランダムな空白の位置情報を作り出す．もし，ランダムな場所を生成する際に，場所が被った場合は continue 処理を行い．再度，位置情報の生成を行う．

2.2　仕様 2 の実現方法

タイルが消える条件は以下の 11 通りのみである．その中で①，②と③，④，⑤，⑥は排反であるが，それ以外のケースの全てに含まれている．よって，より多くのタイルを消すことのできるケースから徐々に場合分けを行い，最終的な消えるタイルを確定させるほうが計算回数が少ない．また，このとき 4 つ消える場合と 3 つ消える場合はもうこれ以上消えるタイルは存在しないので return 文で処理を終わらせる事により，余計な計算をふせいでいる．

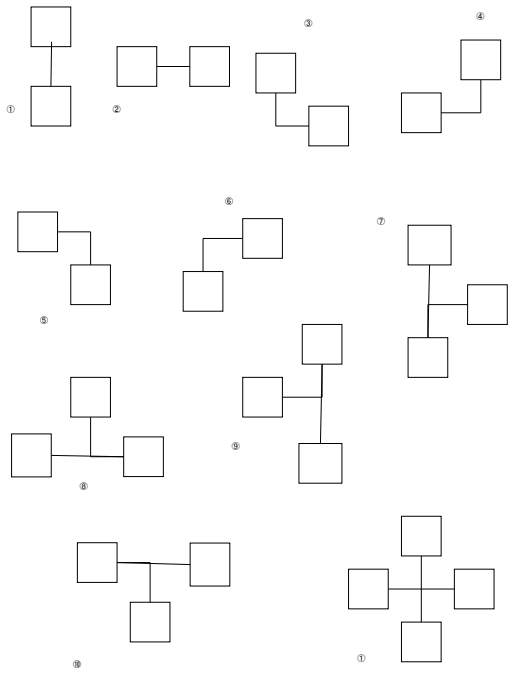

図 1　タイル状態の列挙

2.3　仕様 3 の実現方法

2.2 に記したように，いくつかの場合に分けられた処理のうち，タイルを落とせる分だけスコアを上昇させる．例えるのならば，図 1 の⑦ような形状の場合は 3 枚のタイルを落とすことが可能であるため，スコアを +3 させる．

2.4　仕様 4 の実現方法

この仕様は 2.1 と同じ処理を行った．右クリックが押された際は仕様 1 の関数である initTable() 関数を用いている．

2.5　仕様 5 の実現方法

この仕様実現のために行ったこととして以下がある．

(1). 空白の探索

まず，全体の探索をしている最中に空白をがあれば処理を開始するようにする．これにより，初期状態を判定する際の処理時間が大幅に減少する．また，この探索中に空白状態のタイルをカウントし，その数が縦と横の積と同値であれば盤面全体が空白である．即ち勝利であるため，最後にその処理を追記している．

(2). 横方向の探索

次に，横方向の探索を行う．このとき左右に探索を行い，左右の色のあるタイルにぶつかるまで処理を続ける．左右に色のあるタイルにぶつかるか配列の上限まで探索する．その発見したタイルが同じ色であるかを判定する．このとき，同じ色であった場合は即座に処理を return 文でまだ開けることのできるタイルがあるという返り値を出す．

(3). 縦方向の探索

次に，縦方向の探索をおこなう．このとき上下に探索を行い，上下に色のあるタイルにぶつかるか配列の上限まで探索する．その発見したタイルが同じ色であるかを判定する．このとき，同じ色であった場合は即座に処理を return 文でまだ開けることのできるタイルがあるという返り値を出す．

(4). 上下方向 + 左右方向の探索

正確にはこれは探索ではなく，(3)，(2) で発見した上下方向のタイルと左右方向のタイルを比べて同じものであれば return 文でまだ開けることのできるタイルがあるという返り値を返す．

これらのどれにも当てはまることのない場合は消せるタイルがないということであるため，ゲームを終われせスコアを表示させる．

まとめと反省

まとめ

以下のように今回は探索において場合分けと探索結果の利用をし，枝刈りと計算回数を小さくする工夫を行った．しかし，それ故にいくつかの問題点も発生している．それは次節の反省部分において記す．今回の探索は上下左右に探索をしなければならなったため，必然的に多くの変数を用いてる．今回，変数を減らす工夫も行ったが極端に変数を減らすということには成功していない．

反省

まず，最も大きな問題は今回のコードが非常に冗長になってしまったことだ．同じような処理を何度も繰り返し実行してる．これにより，保守性が著しく低いコードとなってしまった．しかし，これをうまく回避しながら正確なプログラムを書くことは非常に難しかったため断念してしまった．関数等を増やしてもっと単純なコードを目指すべきだったのかもしれない．

結論

今回のカラータイルのプログラムは盤面の探索をし，それによって得たタイルの情報をタイル消去の条件から枝刈りを行うという工夫を行って作成した．

参考文献