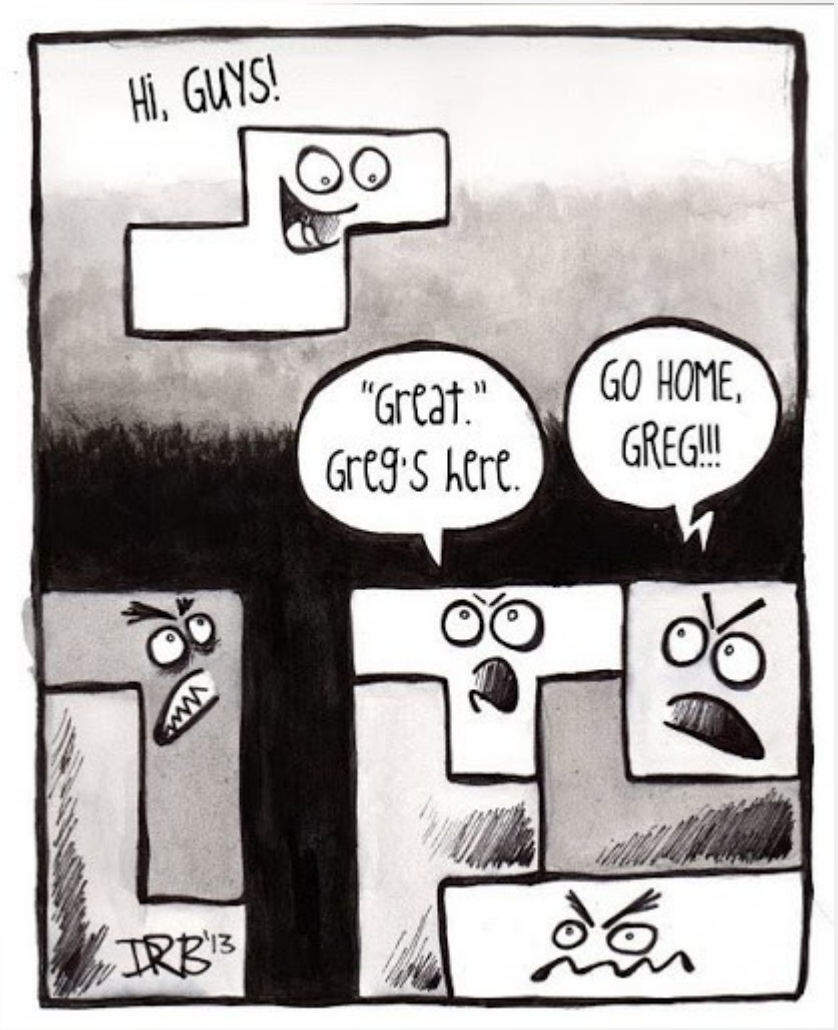
Competitive Tetris Bot

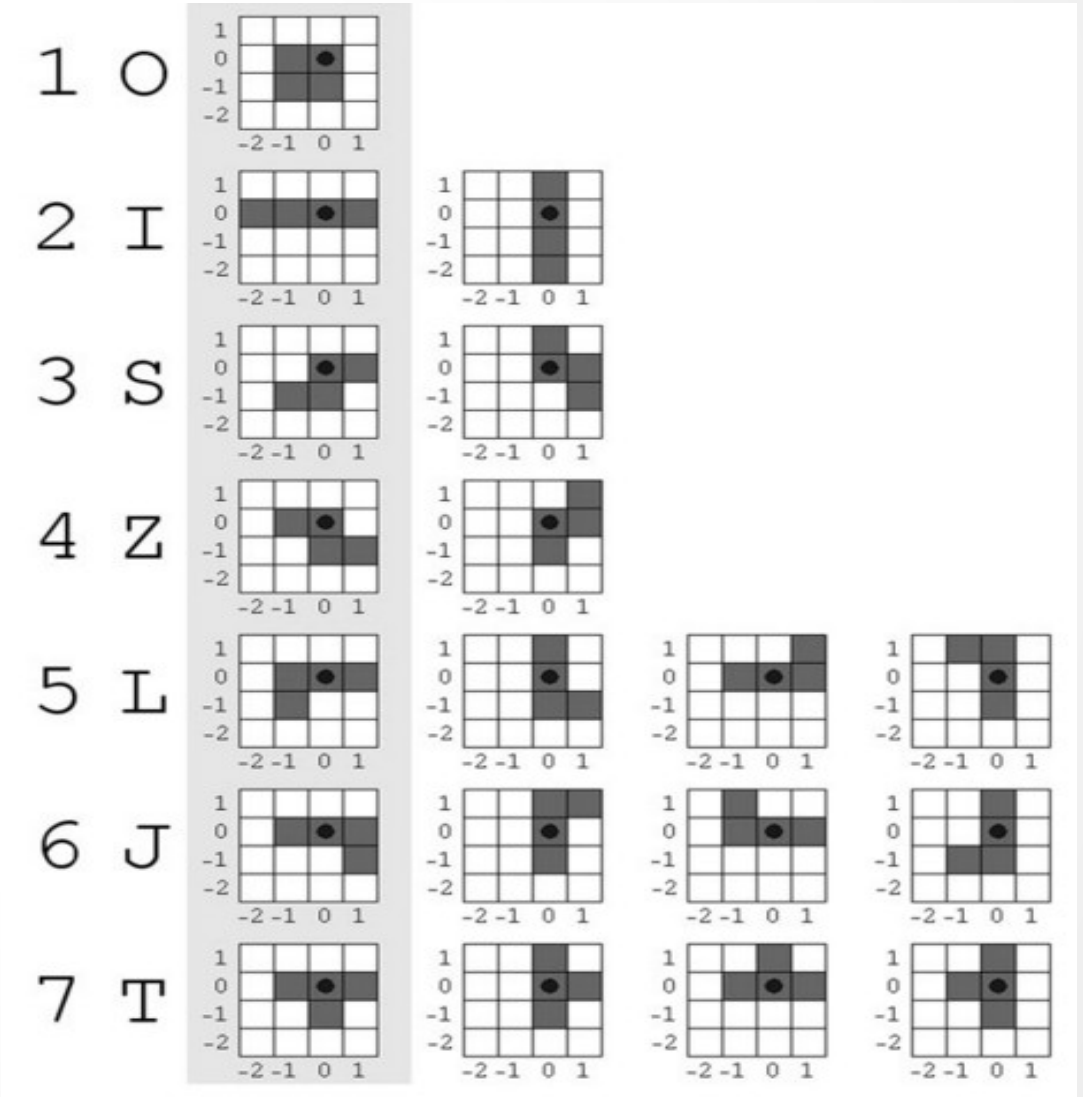
Group 17:
Cem Yusuf Aydoğdu
İrem Ertürk

İstanbul Technical University
April 26, 2016



Tetris Pieces

- There are seven (7) standard Tetris pieces, with the following letter names:
  { O, I, S, Z, L, J, T }
- The letter names are inspired by the shapes of the pieces.

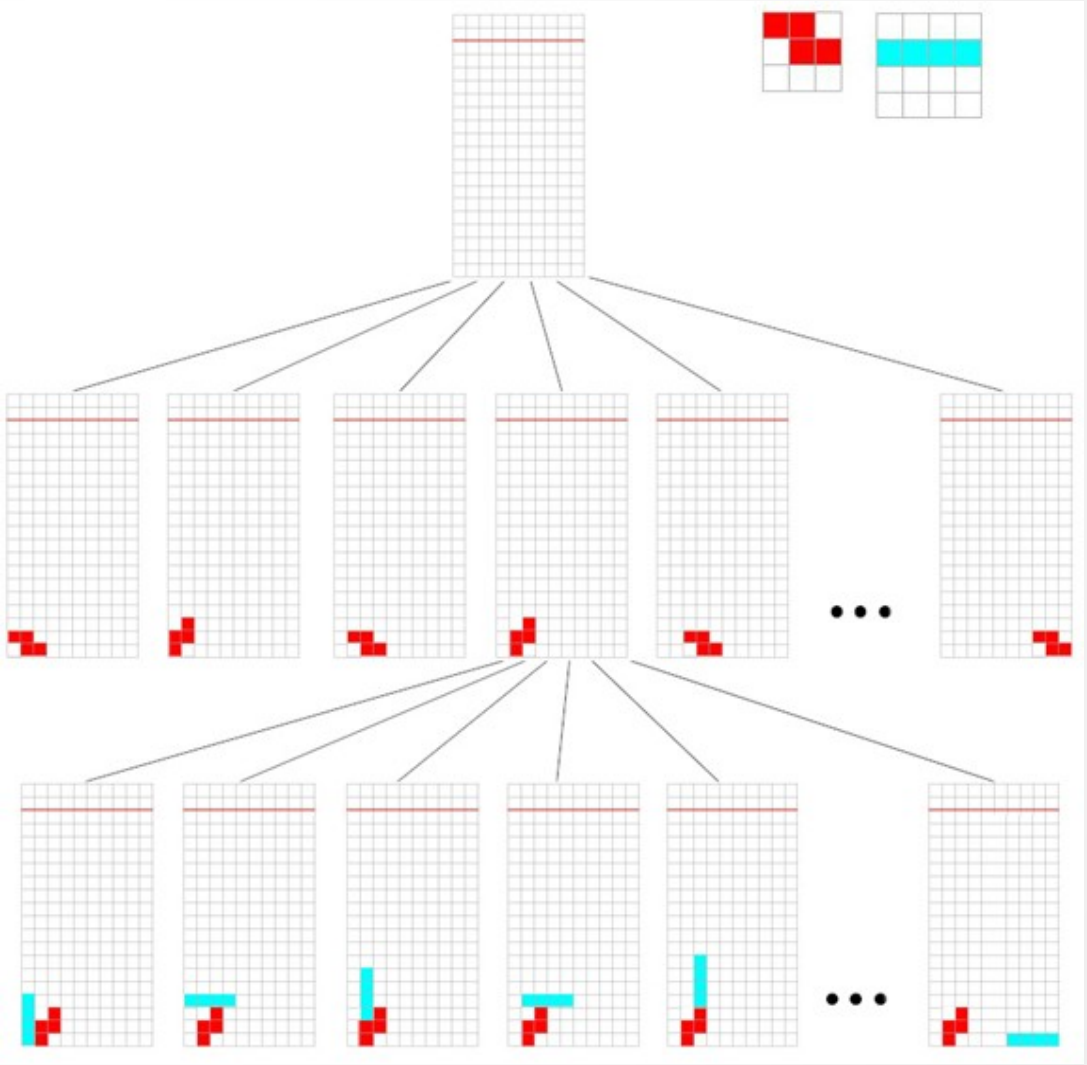


Basic algorithm based on DFS

Visualization of the basic algorithm to find best placement

of pieces,

based on depth first search



Pseudocode for movement planning

Algorithm 1 Algorithm to plan movements according to best score

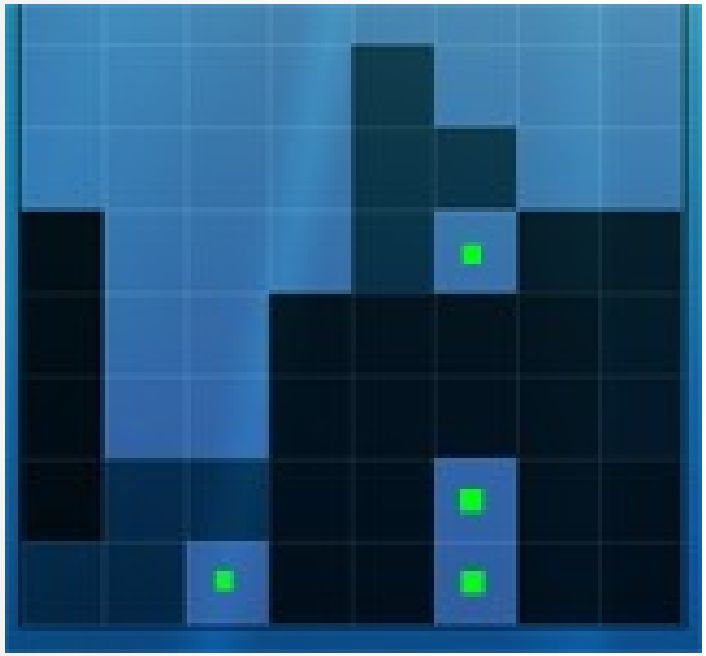
```
1:  procedure GETBESTSCORE(field, piece, nextShape)
2:      maxScore ← −∞
3:      Initialize score, nextScore
4:      for each rotation of current piece do
5:          Move the piece to the leftmost of field
6:          while Shifting the piece to right of field one by one do
7:              Consider placing the piece down
8:              score ← calculateScore()
9:              if nextShape is known then
10:                 nextScore ← getBestScore()        ▷ recursion for next piece
11:             end if
12:             score ← score + nextScore
13:             if score > maxScore then
14:                 maxScore ← score
15:                 Save current movement and rotation counts
16:             end if
17:         end while
18:     end for
19:     return movement, rotation counts of best score
20: end procedure
```

Pseudocode for calculating score

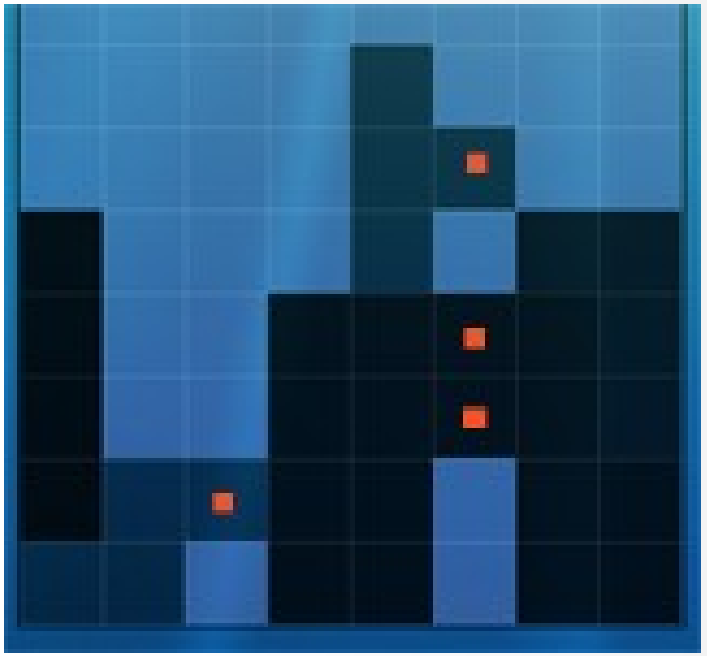
Algorithm 2 Algorithm to calculate scores

1: **procedure** CALCULATESCORE($field$,$piece$)
2: Initialize $score \leftarrow 0$
3: Obtain features F
4: Calculate weights of features, W
5: $score = \sum_{i=1}^{n} F_i \times W_i$
6: **return** $score$
7: **end procedure**

Scoring Features



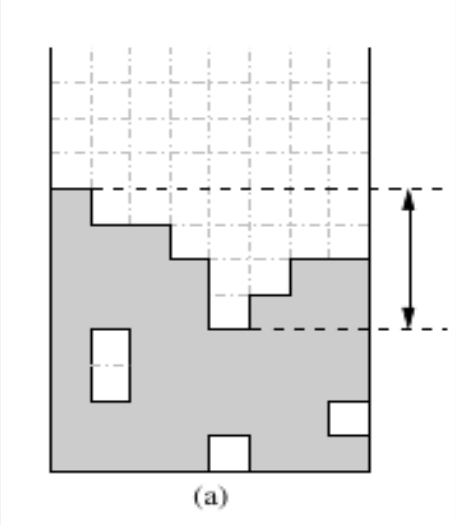
Holes



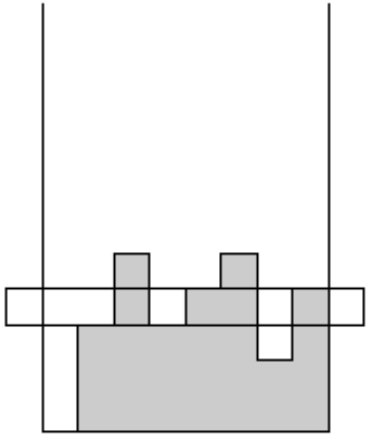
Blockades

https://luckytoilet.wordpress.com/2011/05/27/coding-a-tetris-ai-using-a-genetic-algorithm/

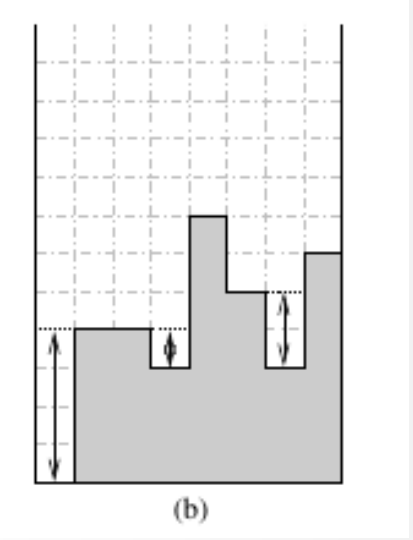
Scoring Features



Altitude difference



Row transitions



Wells

Böhm, Kókai, Mandl, “An evolutionary approach to tetris” presented as Sixth Metaheuristics International Conference, 2005, p. 5.

Score Evaluation

	Features	**Weigts**
1	Max height	-6.2709
2	Number of holes	-7.0271
3	Number of blockades	-7.0271
4	Count of removed lines	7.8621
5	Altitude difference in the field	-8.5395
6	Number of wells	-3.381
7	Number of row transitions	-8.4262
8	Number of column transitions	-10.4262
9	Height of new placed piece	-8.50

Implementation Details

```java
public double calculateScore(Shape piece, int combo, int round) {

    double score = 0;

    scores.calculateFeatures();

    // Additional control statements
    double removedWeight = 1;
    if (scores.removedLines == 1) {
        if (round < 30) // Below round thresold, try to score more
            removedWeight = -2;

    } else if (scores.removedLines == 0)
        removedWeight = combo + 1;
    else
        removedWeight = (scores.removedLines + combo) * 2.35;

    double placementHeightWeight = (this.getHeight() - piece.getLocation().getY() - (piece.getSize() / 2));
    ShapeType piece_type = piece.getType();

    if (scores.maxHeight > 12) // If h exceeds 12, try to eliminate more
                               // lines
        placementHeightWeight *= 2;
    if (piece_type == ShapeType.I) // Do not place I to sides
        placementHeightWeight *= 2.5;
    else if (piece_type == ShapeType.S || piece_type == ShapeType.Z)
        placementHeightWeight *= 1.2;

    score += scores.maxHeight * -6.2709 + scores.holes * -7.0271 + scores.blockades * -7.0271
            + removedWeight * 7.8621 + scores.altitudeDifference * -8.5395 + scores.wells * -3.381
            + scores.rowTransitions * -8.4262 + scores.columnTransitions * -10.4262
            + placementHeightWeight * -8.500158825082766;

    return score;
}
```

Thanks for listening..