ECE374 SP23 HW6

Contributors

Zhirong Chen (zhirong4)

Ziyuan Chen (ziyuanc3)

Problem 3

The traditional world chess championship is a match of 24 games. The current champion retains the title in case the match is a tie. Each game ends in a win, loss, or draw (tie) where wins count as 1, losses as 0, and draws as $\frac{1}{2}$.

The players take turns playing white and black. White plays first and has an advantage. The champion plays white in the first game. The champion has probabilities ww, wd, and wl of winning, drawing, and losing playing white, and has probabilities bw, bd, and bl of winning, drawing, and losing playing black.

(a) Write a recurrence for the probability that the champion retains the title. Assume that there are g games left to play in the match and that the champion needs to get i points (which may be a multiple of $\frac{1}{2}$).

(b) Based on your recurrence, give a dynamic programming algorithm to calculate the champion's probability of retaining the title.

(c) Analyze its running time for an n-game match.

Solution

(a)

As a starting point, we assume that the champion plays white in the first game.

```
float ww, wd, wl, bw, bd, bl;   // given constants

float probability(int g, float i) {
    if (i > 0)
        if (g > 0)
            if (g % 2 == 0)     // this condition may be revised
                return ww * probability(g-1, i-1)
                     + wd * probability(g-1, i-0.5)
                     + wl * probability(g-1, i);
            else
                return bw * probability(g-1, i-1)
                     + bd * probability(g-1, i-0.5)
                     + bl * probability(g-1, i);
```

```
        else
            return 0;           // chances used up
    else
        return 1;               // score cannot be negative
}
```

Since i is discrete (must be a multiple of $\frac{1}{2}$), we can use a 2D array $M \in \mathbb{R}^{g \times 2i}$ to memoize the results of recursive calls. This is an important optimization concerning time complexity.

(b)

The champion needs to take at least 12 points, so

```
p = probability(24, 12, ww, wd, wl, bw, bd, bl)
```

(c)

Without memoization, time complexity can be illustrated as a tree:

```
          n
       /  |  \
   n-1   n-1   n-1
  / | \ / | \ / | \
```

Each node take $O(1)$ to compute, so total cost is $1 + 3 + \ldots + 3^n = O(3^n)$

But with memoization, we can optimize the time complexity to $O(n^2)$, or $O(gi)$ where $g = n$ and $i = 2n$.