CS390-ATA: Homework 2

Jayden Lee, Saad Sharief

Due Date: **February 06, 2025**

Contents

Exercise 5.10

We will consider the game of Set as an optimization problem where we also know the full sequence of cards in the deck. The three decks are represented by three arrays $A[1..m]$, $B[1..n]$, and $C[1..p]$. The cards are listed from top to bottom in each deck. For simplicity, we assume access to a constant-time subroutine `IsItASet(i,j,k)` that takes as input three indices i, j, k and outputs *True* if the cards $(A[i], B[j], C[k])$ form a set, and *False* if they do not.

The problem is to compute the maximum number of sets that can be obtained in a game of Solitaire Set over the decks $A[1..m]$, $B[1..n]$, and $C[1..p]$. Design and analyze an algorithm for this problem.

1. **Recursive Specification**

   `set-solitaire(i, j, k)`: returns the maximum number of sets given three decks of cards represented by $A[i..m]$, $B[j..n]$ and $C[k..p]$.

2. **Recursive Implementation**

   `set-solitaire(i, j, k)`

   1. **If** $i > m$ or $j > n$ or $k > p$ or $i + j + k < 3$ **then return** 0
   2. **If** `IsItASet`$(A[i], B[j], C[k])$ is *true* **then**
      - **return** $\max\{1 +$ `set-solitaire`$(i+1, j+1, k+1)$,
        `set-solitaire`$(i+1, j, k)$,
        `set-solitaire`$(i, j+1, k)$,
        `set-solitaire`$(i, j, k+1)\}$
   3. **Else**
      - **return** $\max\{$ `set-solitaire`$(i+1, j, k)$, `set-solitaire`$(i, j+1, k)$, `set-solitaire` $(i, j, k+1)\}$

3. **Solving the original problem**

   The maximum of `set-solitaire(i, j, k)` over all $i \in [m]$, $j \in [n]$, and $k \in [p]$ gives the number of the maximum number of sets.

4. **Runtime analysis**

   The runtime of this algorithm run naively should be exponential in the order of $O(4^{m+n+p})$. However, by caching and reusing computations, we can essentially bring it down to the number of subproblems we have to solve–one for each combination of indices. Therefore, the time complexity of this algorithm is

   $$T(n) = O(mnp) \tag{1}$$

5. **Proof**

We prove that `set-solitaire(i,j,k)` returns the maximum number of sets possible from the decks $A[i..m]$, $B[j..n]$, and $C[k..p]$. We use induction on $m - i + n - j + k - p$ (the total number of remaining cards).

In the base case, when $m - i + n - j + k - p < 3$ or any index exceeds its deck size ($i > m$, $j > n$, or $k > p$), there aren't enough cards to form any sets, so returning 0 is correct.

For the inductive step, assume the claim holds for all positions with fewer total remaining cards. Consider position (i, j, k) with cards $A[i]$, $B[j]$, and $C[k]$ at the tops of the decks.

Let OPT be an optimal sequence of moves from this position. We have two main cases:

Case 1: If $A[i]$, $B[j]$, $C[k]$ form a set

OPT can either:

- Take this set and continue optimally with the remaining cards, giving $1 + \text{set-solitaire}(i + 1, j + 1, k + 1)$ sets
- Skip $A[i]$ and continue optimally, giving $\text{set-solitaire}(i + 1, j, k)$ sets
- Skip $B[j]$ and continue optimally, giving $\text{set-solitaire}(i, j + 1, k)$ sets
- Skip $C[k]$ and continue optimally, giving $\text{set-solitaire}(i, j, k + 1)$ sets

By the inductive hypothesis, each recursive call gives the optimal number of sets for its subproblem. Therefore, taking the maximum of these options gives the optimal number of sets possible from position (i, j, k).

Case 2: If $A[i]$, $B[j]$, $C[k]$ don't form a set

Since these cards can't form a set, OPT must skip at least one of them. Therefore, OPT must achieve its value by either:

- Skipping $A[i]$ and continuing optimally, giving $\text{set-solitaire}(i + 1, j, k)$ sets
- Skipping $B[j]$ and continuing optimally, giving $\text{set-solitaire}(i, j + 1, k)$ sets
- Skipping $C[k]$ and continuing optimally, giving $\text{set-solitaire}(i, j, k + 1)$ sets

Again, by the inductive hypothesis, each recursive call gives the optimal value for its subproblem. Taking the maximum of these options gives the optimal number of sets possible from position (i, j, k).

For both cases, the recursive implementation considers all possible optimal choices and returns the maximum value among them. Therefore, $\text{set-solitaire}(i, j, k)$ returns the optimal number of sets possible from position (i, j, k).

This completes the proof that the recursive specification is satisfied for all valid indices i, j, and k.