Kick Start 2018 - Round H

Analysis: Let Me Count The Ways

Let Me Count The Ways: Analysis

Small dataset

We can solve the Small dataset using dynamic programming. Let *f(x, y, z)* (*x* and *y* are non-negative integers, while *z* is 0 or 1) be the number of ways to arrange a seating order for *x* non-newlywed people and *y* newlywed couples (thus the total is *x* + 2*y* people and seats), such that:

- There is no newlywed couple for which the two members are seated next to each other, and
- If *z* = 1, then one of the *x* non-newlywed people must not sit in the first remaining unassigned seat.

The base case of this function occurs when there are no people left (i.e. *x* = *y* = 0), then *f(x, y, z)* = 1. Otherwise, *f(x, y, 0)* can be defined recursively as follows:

- We can put one of the *x* non-newlywed people in the first seat. Therefore, we need to assign the remaining *x* - 1 non-newlywed people and *y* newlywed couples. This contributes *x* × *f(x - 1, y, 0)* to the total number of ways captured by *f(x, y, 0)*.
- We can put one of the member of the *y* newlywed couples in the first seat. The other member of this couple can be considered as an additional non-newlywed person. Therefore, we need to assign the remaining *x* + 1 non-newlywed people and *y* - 1 newlywed couples. However, note that this additional person cannot be seated next to his/her newlywed partner (i.e. cannot be seated on the first unassigned seat after his/her newlywed partner sit down). Therefore, this contributes 2 × *y* × *f(x + 1, y - 1, 1)* to the total number of ways captured by*f(x, y, 0)*.

Therefore, *f(x, y, 0) = x* × *f(x - 1, y, 0)* + 2 × *y* × *f(x + 1, y - 1, 1)*. The cases for *f(x, y, 1)* are similar, except that we can't put one of the *x* non-newlywed people in the first seat. Therefore, *f(x, y, 1)* = (*x* - 1) × *f(x - 1, y, 0)* + 2 × *y* × *f(x + 1, y - 1, 1)*.

This dynamic programming solution runs in O(**N** × **M**) space and time.

Large dataset

We can solve the Large dataset using the inclusion-exclusion principle. Since **M** can be up to 100000, it is infeasible to iterate all possible subset of the newlywed couples. However, we can observe that the number of ways to arrange a seating order such that the newlywed couple P sits adjacently (i.e. the two members of a newlywed couple P sit next to each other) is the same as the number of ways to arrange a seating order such that the newlywed couple Q sits adjacently. In general, for a fixed number *k* of newlywed couples, regardless of which particular couples we choose, the number of seating orders in which all of those *k* newlywed couples have their two members adjacent is the same.

Therefore, we can define a function *g(k)* as the number of ways to arrange a seating order such that for each of the first *k* newlywed couples, the two members of a couple are sitting next to each other. *g(k)* can be computed by multiplying the following:

- The number of ways these newlywed couples sit. Since the members of each newlywed couple must sit next to each other, we can treat each newlywed couple as if it were one

person. Therefore, there are $2\mathbf{N} - k$ seats for k couples. Since the couples are not identical and the order of the couples matters, there are $C(2\mathbf{N} - k, k) \times k!$ ways, where $C(n, k)$ is the number of ways of picking k unordered outcomes from n possibilities (i.e. "n choose k").
- The number of ways of ordering two people in each newlywed couple, which is 2^k.
- The number of ways in which the remaining people can sit, which is $(2(\mathbf{N} - k))!$.

Therefore, $g(k) = C(2\mathbf{N} - k, k) \times k! \times 2^k \times (2(\mathbf{N} - k))!$

Finally, we should not forget that $g(k)$ counts the number of ways to arrange a seating order such that the first k newlywed couples sit adjacently. To consider all subset of newlywed couples of size k, we need to multiply $g(k)$ by $C(\mathbf{M}, k)$.

Putting all the pieces together, the answer we are looking for is $\Sigma\ (-1)^k \times g(k) \times C(\mathbf{M}, k)$ for all $0 \le k \le \mathbf{M}$. By precomputing the value of powers of two, factorials, and their modular inverses, this answer can be computed in $O(\mathbf{N} + \mathbf{M})$ space and time.