

HCPC 新歓2022

Box and Ball(Diff: 981)

解説：Slephy

問題文（引用）

- N 個の箱があります。箱は 1 から N まで番号が振られています。最初、1 番目の箱には赤いボールが 1 個入っています。また、2～ N 番目の箱には白いボールが 1 個ずつ入っています。
- 高橋君は M 回の操作を順に行います。 i 回目の操作では、 x_i 番目の箱から適当なボールを 1 個選び、それを y_i 番目の箱へ移します。
- 高橋君がすべての操作を終えた後、赤いボールが入っている可能性のある箱は何個か求めてください。

まずは問題文を理解しよう

入力例 1

3 2

1 2

2 3

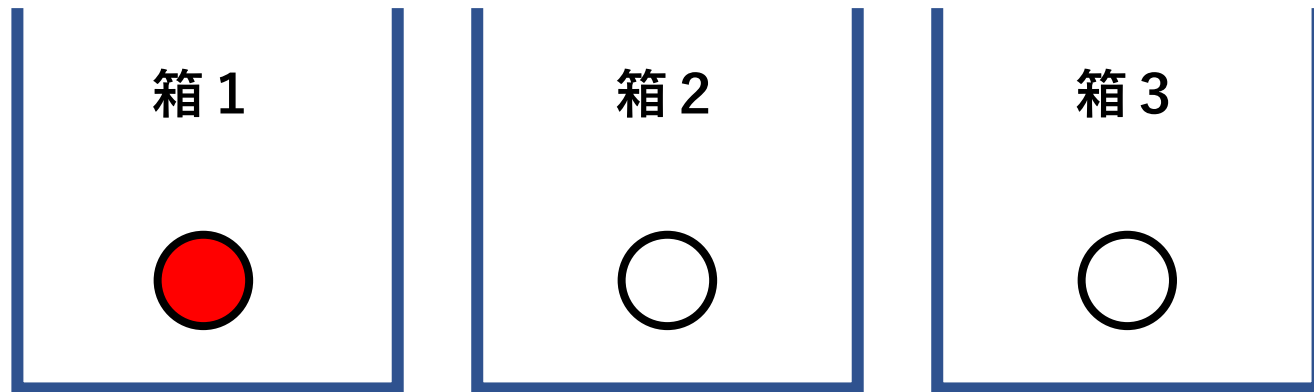


$N = 3$

$M = 2$

$(x_1, y_1) = (1, 2)$

$(x_2, y_2) = (2, 3)$



箱が3個

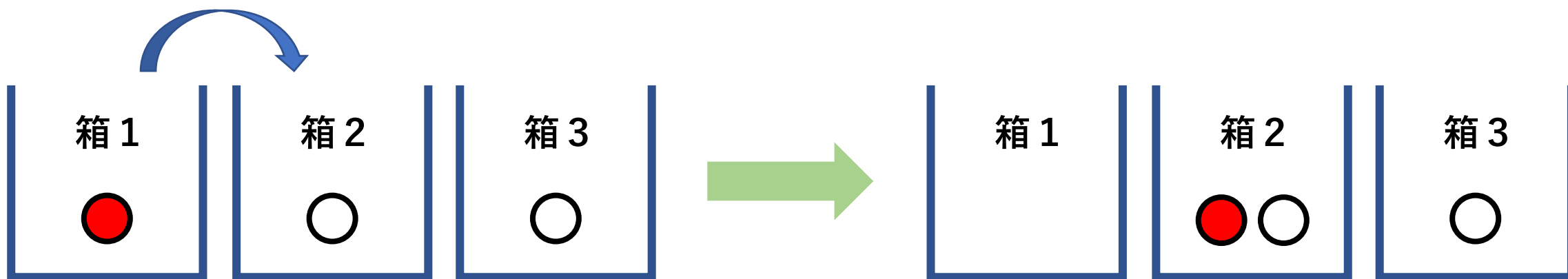
ボールの移動は2回

移動1回目: 箱1→箱2

移動2回目: 箱2→箱3

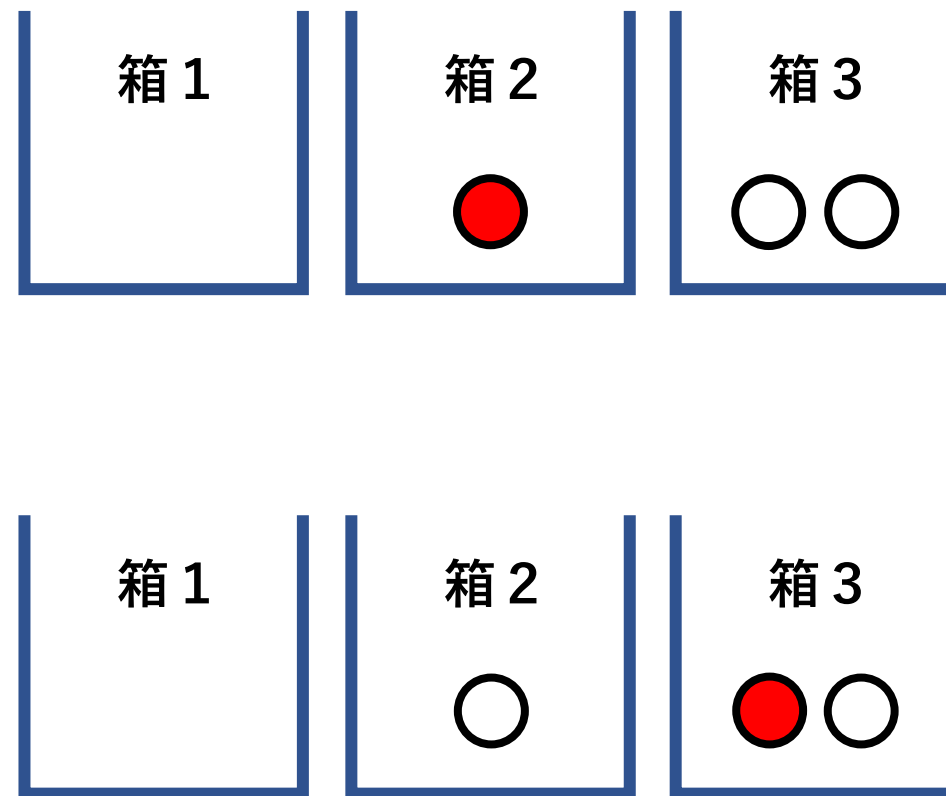
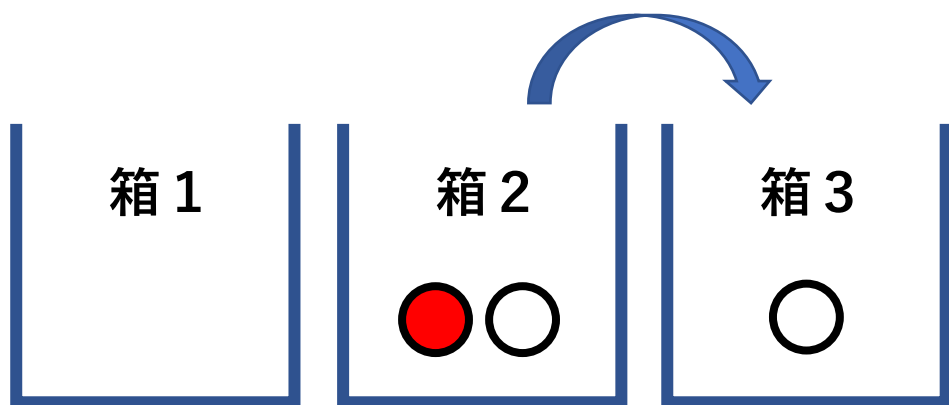
移動1回目

箱1→箱2

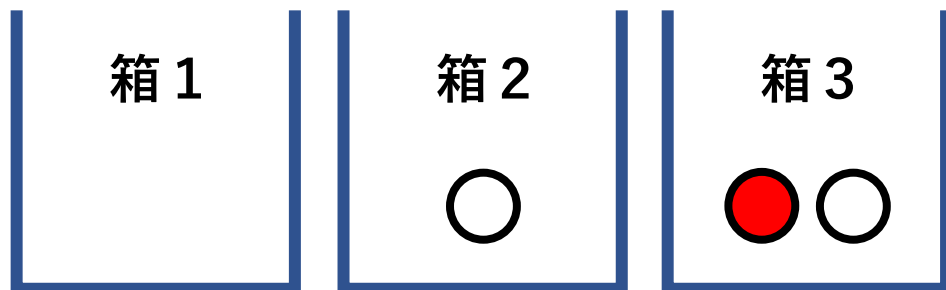
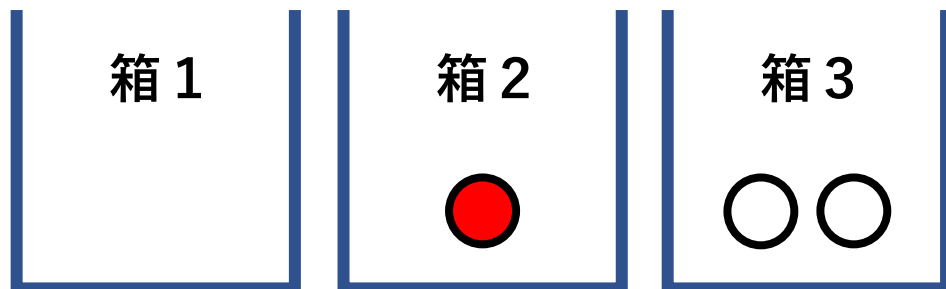


移動2回目

箱2→箱3



このケースの答え



最終的に考えられる状態は左に示す2つのみ。
赤いボールが入っている可能性がある箱は
箱2と箱3の二つ。

$\therefore \text{Answer} = 2$

ここからが本題

この問題の難しさ

制約が大きい！

$$2 \leq N \leq 10^5, \quad 2 \leq M \leq 10^5$$

先ほどのような終了状態を全通り列挙しては到底間に合わない
なんと指数オーダー $O(2^N)$ かかってしまう

何とか間に合わないか…？

- ・ 同じ色のボールを区別せずに処理してみたら？

結局**指数オーダー**…

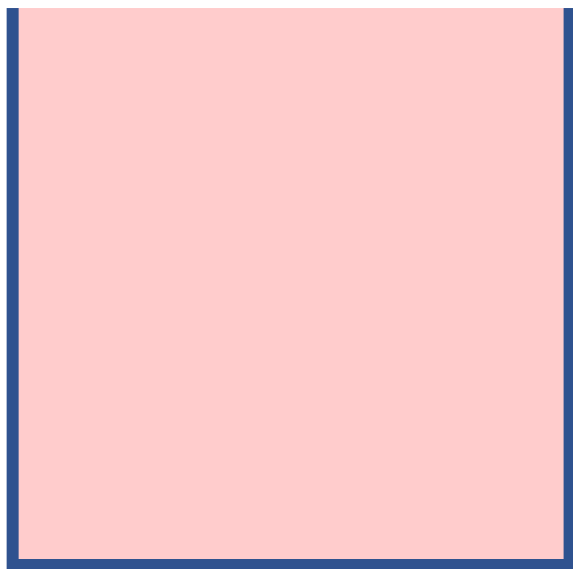
ボトルネックになるのは、**取りうる状態数の多さ**。

解法のポイント

- 最終的に問題になるのは
「その箱に赤いボールが入っている可能性があるか否か」
- 現在どの箱に赤いボールが入っているかを気にせず、
「赤いボールが入っている可能性」の遷移に着目して考える
- これにより、 i 回操作が終わった時点で有り得るすべての状態をまとめて表現できる（うれしい）

まずは定義

赤いボールが入っている可能性がある箱

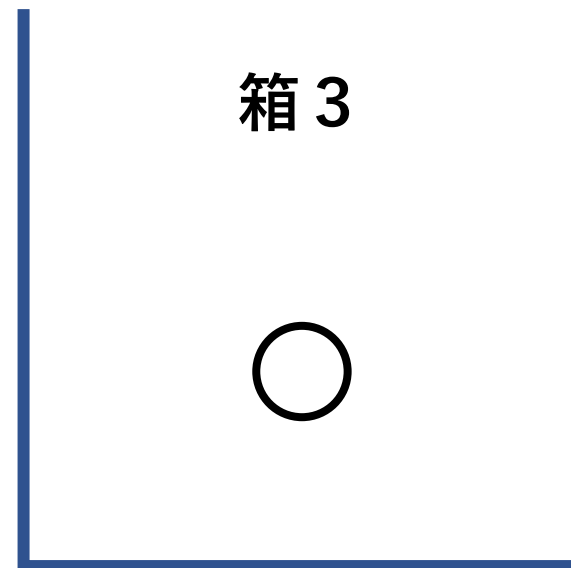
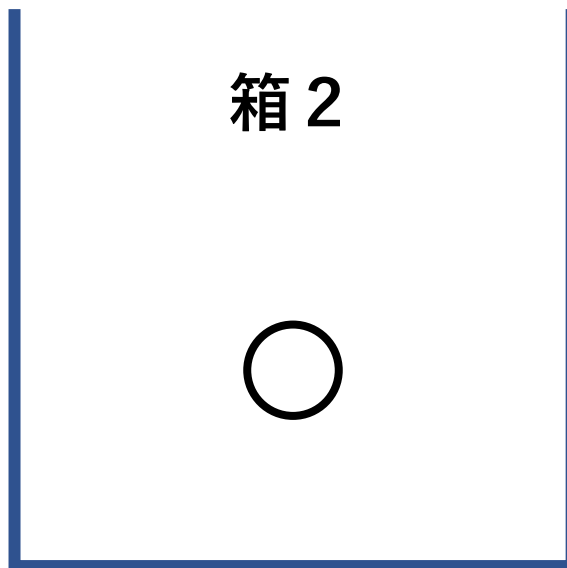
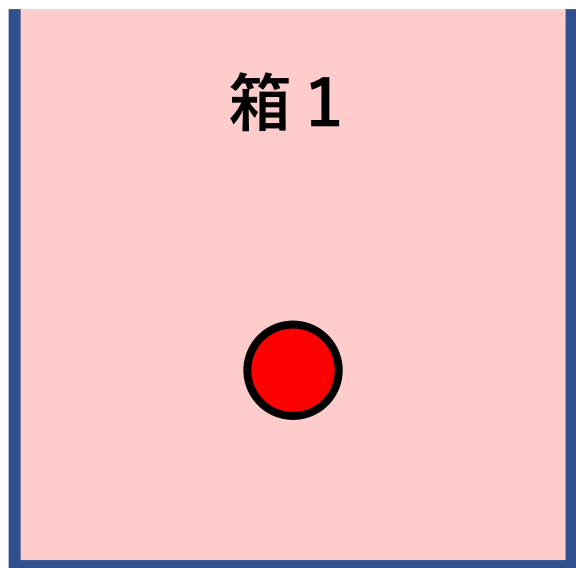


〃可能性がない箱



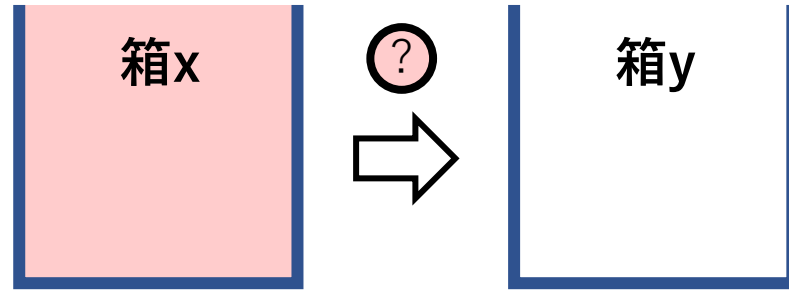
以下、「赤いボールが入っている可能性」を単に「可能性」と書きます。

初期状態における可能性

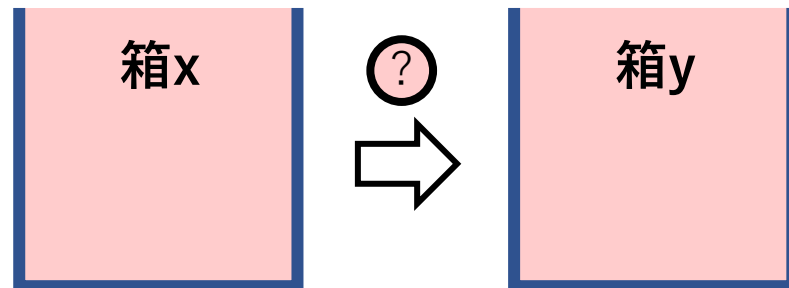
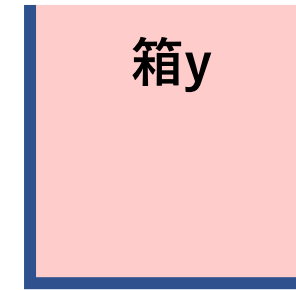


可能性の遷移（移動先の箱①）

「可能性」がある箱からの移動



移動後



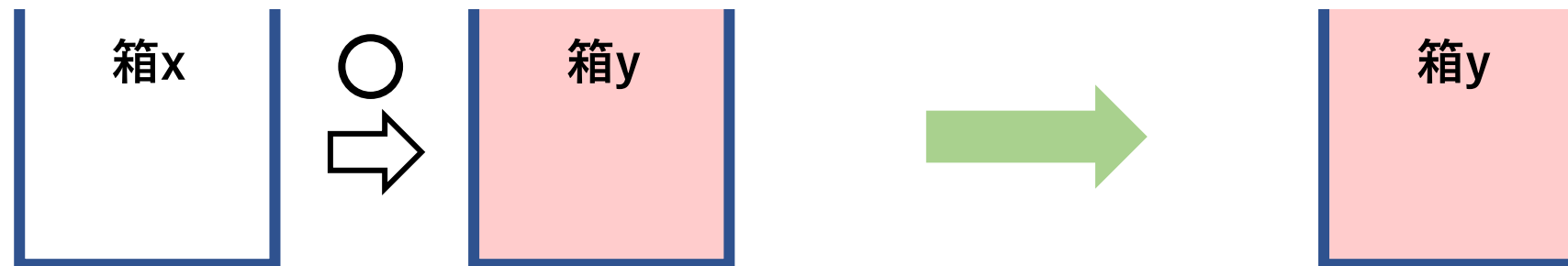
移動後



可能性が
伝播する

可能性の遷移（移動先の箱②）

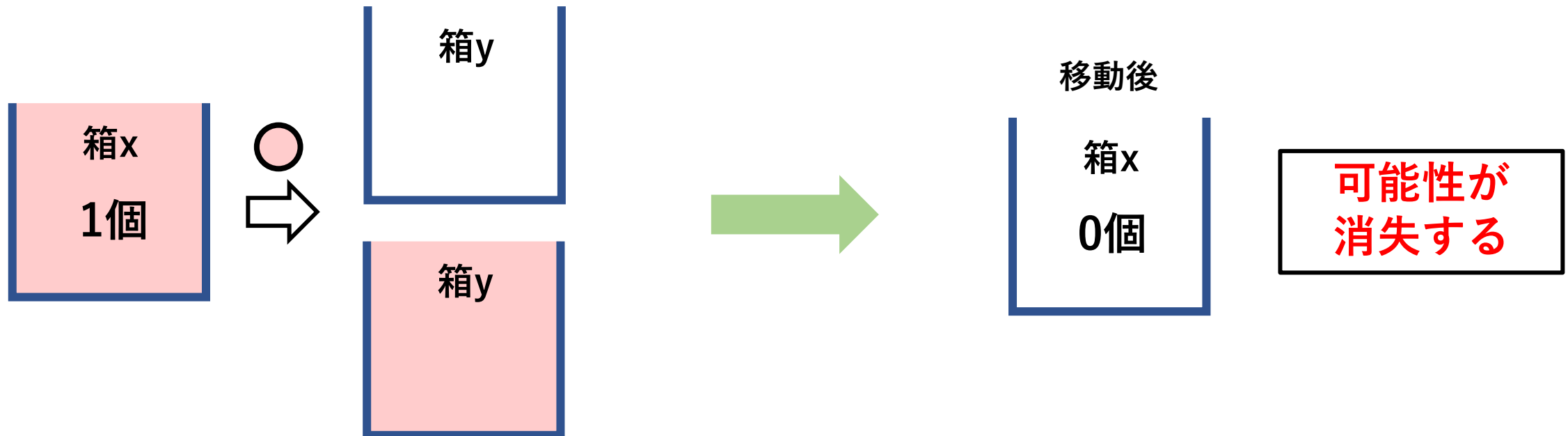
「可能性」がない箱からの移動



可能性が
伝播しない

可能性の遷移（移動元の箱）

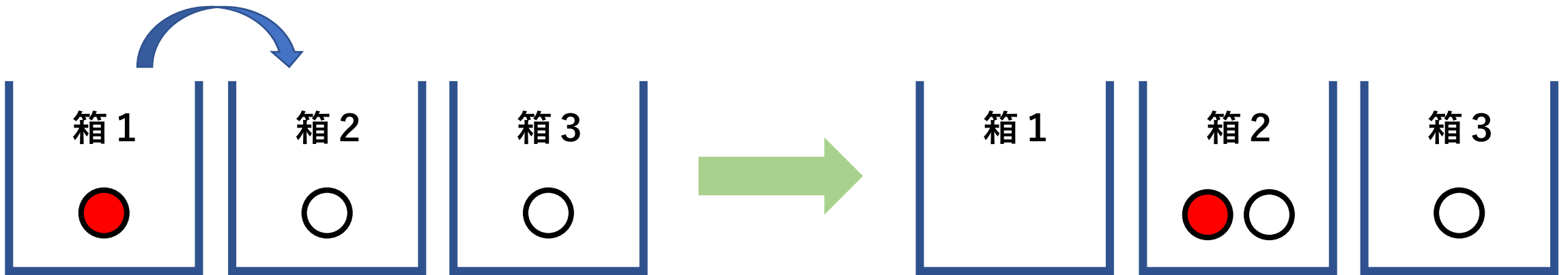
基本的に不変



実際に適用してみよう！

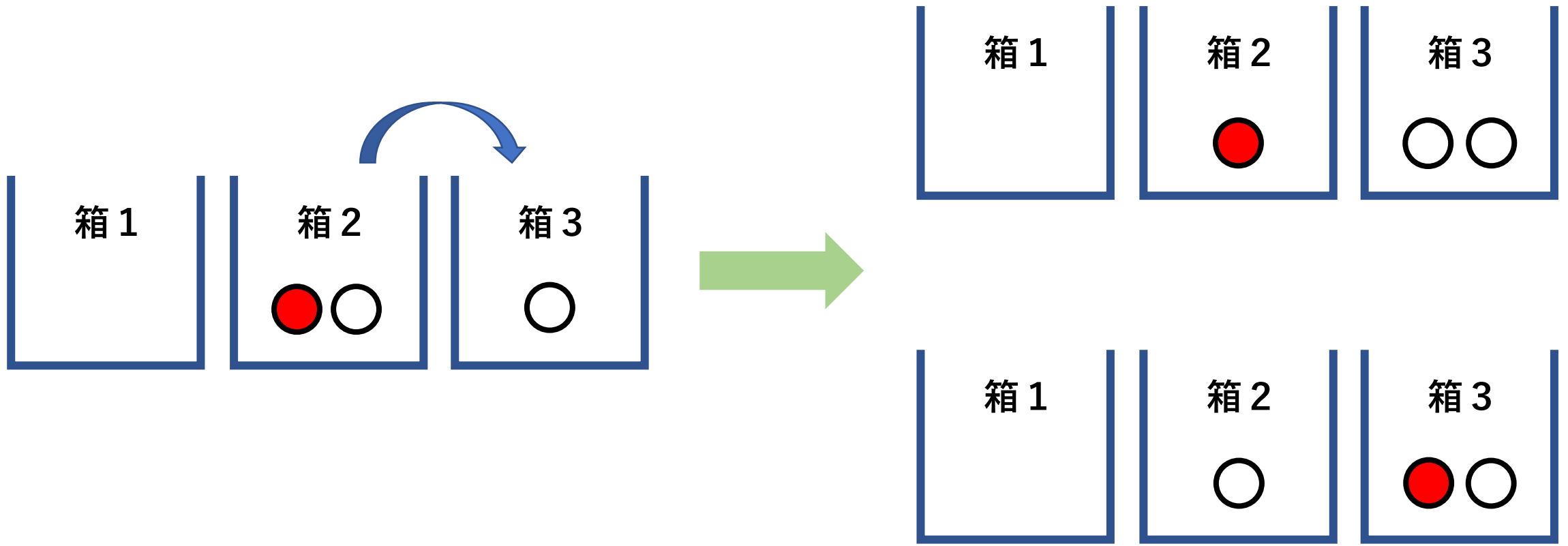
移動1回目（従来の方法）

箱1→箱2



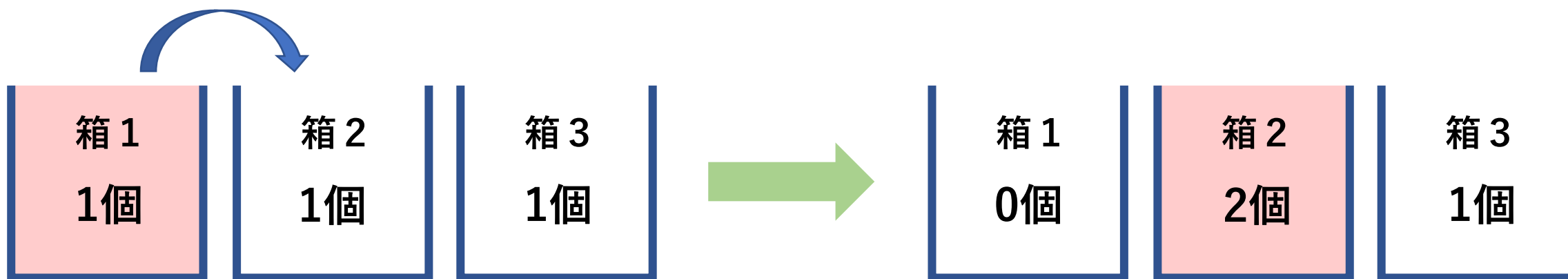
移動2回目（従来の方法）

箱2→箱3



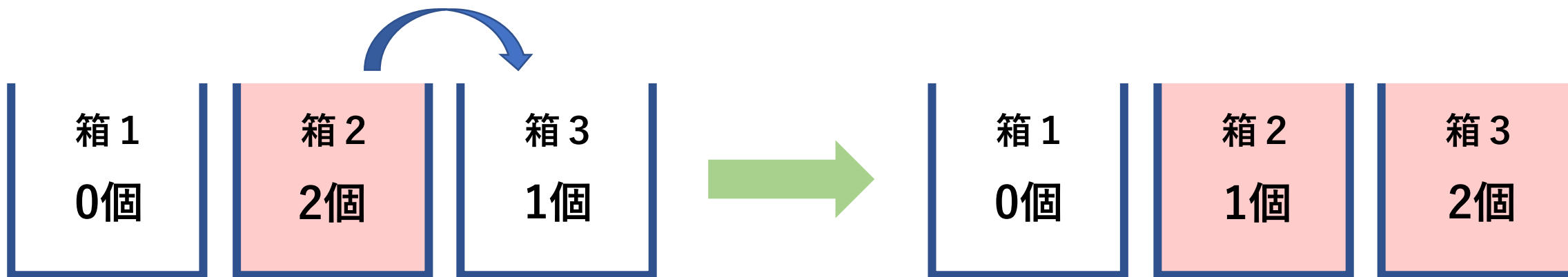
移動1回目（想定解）

箱1→箱2



移動2回目（想定解）

箱2→箱3



2種類の状態をひとつにまとめて
表すことができる！

このアルゴリズムの計算量

- 1回の操作は定数時間で終わる。 $O(1)$
- それがM回あるので、 $O(M)$
- 配列の初期化に $O(N)$ かかるので、**合計で $O(N+M)$**
- $2 \leq N \leq 10^5$, $2 \leq M \leq 10^5$ なので間に合う！

ACコード例

```
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 int main(){
5     // 入力
6     int n, m; cin >> n >> m;
7
8     // 配列の初期化(0-indexed)
9     vector<int> ballNum(n, 1);
10    vector<bool> mayRed(n, false);
11    mayRed[0] = true;
12
13    for(int i = 0; i < m; i++){
14        int x, y; cin >> x >> y;
15        x--; y--; // 0-indexedに変換
16
17        // 移動先の箱へ可能性を伝播させる
18        if(mayRed[x]) mayRed[y] = true;
19
20        // ボールの個数を変更
21        ballNum[x]--;
22        ballNum[y]++;
23        // 移動元の箱が空になったら、可能性をfalseにする
24        if(ballNum[x] == 0) mayRed[x] = false;
25    }
```

```
27    // 赤いボールが入っている可能性がある箱の数を数えて出力する
28    int ans = 0;
29    for(int i = 0; i < n; i++){
30        if(mayRed[i]) ans++;
31    }
32    cout << ans << endl;
33    return 0;
34 }
```

参考提出

- **C++(49ms)**

<https://atcoder.jp/contests/agc002/submissions/33780398>

- **Python(220ms)**

<https://atcoder.jp/contests/agc002/submissions/33767850>