**Coin-Row Problem**

Dr. Byun

09/24/2020

**Problem Definition**

Assume that there are *n* coins on the street in a row.

The problem is to pick up a few coins to get the maximum amount.

One constraint of the problem is that you can’t select two adjacent coins.

In other words, if you choose a coin, you must skip one or more of the next coin(s).

In the problem, we denote the value of each coin by c1, c2, …, cn.

**Example**

Assume that there are three coins like below

3, 7, 5

To get the maximum amount, you will pick the first coin (= value 3) and third coin (= value 5). If you select the second coin (= value 7), you will not be able to select the first coin and the third coin due to the constraint of the problem.

By the way, we represent the value of each coin as follows:

c1 = 3

c2 = 7

c3 = 5

**Exercise**

Assume that there are six coins like below

5, 1, 2, 10, 6, 2

Which coins would you pick to get the maximum amount?

**Solution**

To get the maximum amount, you must select the first coin (= value 5), the fourth coin (= value 10) and the sixth coin (= value 2).

The total amount is 17.

**Dynamic Programming**

To solve the problem, we should set up a recurrence relation. For this, let **F(n)** be the maximum amount for ***n*** coins.

For example, for the three coins example (= 3, 7, and 5), F(3) represents the maximum amount for the three coins. In other words,

F(3) = 8

Now, we will add each coin one by one. In other words, F(0), F(1), F(2), and F(3) are calculated in sequence. To keep the values, we use a table like below. In the table, the first row is the index and the second row is the coin value (= c1, c2, and c3).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 |
| Ci | NA | 3 | 7 | 5 |
| F(i) |  |  |  |  |

As a starting point, let's assume that there are no coins on the street. Since there is nothing to choose from, the maximum amount is 0. We can represent it as

F(0) = 0

Now, if there’s only one coin (= the first coin with value 3) on the ground, we will pick the coin because it guarantees the maximum amount. So,

F(1) = 3

The first two cases (= F(0) and F(1)) are obvious, and we can keep the results in the table below

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 |
| Ci | NA | 3 | 7 | 5 |
| F(i) | **0** | **3** |  |  |

Now suppose that the second coin (= value 7) is added to the problem. If we have a new coin (= the second coin), we have two options. The first option is to pick (or select) the new coin. The second option is to ignore (or deselect) the new coin. This is a detailed description of each option.

1. Pick the last coin (= second coin). If we pick the second coin of the value c2, we can not pick the first coin because it is adjacent to the second coin. So, F(2) is only the value of the second coin (and the empty coin). We can express it as follows. Note that we put F(0) to indicate the empty coin situation.

F(2) = F(0) + c2 = 0 + 7 = 7

1. Ignore the last coin (= second coin). If we do not select the second coin, the amount of this option is equal to F(1). That is, F(2) is the same as F(1).

F(2) = F(1) = 3

Because 7 is larger than 3 between the two options, we will choose the first option. So we know that F(2) is 7 and keep the information to the table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 |
| Ci | NA | 3 | 7 | 5 |
| F(i) | 0 | 3 | **7** |  |

So far, we calculated F(0), F(1), and F(2). Now, suppose that the third coin (= value 5) is added to the problem. Again, we have two options for the new coin (= the third coin). The first option is to pick (or select) the new coin. The second option is to ignore (or deselect) the new coin. This is a detailed description of each option.

1. Pick the last coin (= third coin). Choosing the third coin means that the second coin is not selectable because it is adjacent to the third coin. Therefore, in this option, the value of F(3) is the sum of the third coin value (= c3) and F(1). We can express it as follows. Note that we put F(1) to indicate the maximum amount to the first coin.

F(3) = F(1) + c3 = 3 + 5 = 8

1. Ignore the last coin (= third coin). If we do not select the third coin, the amount of this option is equal to F(2). We can represent it as

F(3) = F(2) = 7

Because 8 is larger than 7 between the two options, we will choose the first option. So we know that F(3) is 8 and keep the information to the table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 |
| Ci | NA | 3 | 7 | 5 |
| F(i) | 0 | 3 | 7 | **8** |

Now, we finished the calculation of F(3) and will pick the first and third coins to get the amount (= 8).

For the general *n* coins we can represent *F(n)* in the following recurrence.

F(n) = max {F(n-2) + cn, F(n-1)} for n > 1

The formula indicates that F(n) will be the maximum value between two options:

F(n-2) + cn: Pick the last coin (= value cn). Selecting the last coin means that you can not pick the second to the last coin because it is adjacent to the last coin. Since we represent the maximum amount from the first coin to (n-2)th coin as F(n-2), the total value of this option is F(n-2) + cn.

F(n-1): Ignore the last coin. Deselecting the last coin means that F(n) is the same as F(n-1) (= maximum amount from the first coin to (n-1)th coin).

In addition to the recurrence, we have two obvious initial conditions

F(0) = 0

F(1) = c1

**Exercise**

Assume that there are six coins like below

5, 1, 2, 10, 6, 2

Solve the problem using the dynamic programming.

**Solution**

Our goal is to calculate F(6). To solve it, we should calculate F(0), F(1), F(2), … F(6) one by one. Figure 8.1 (text page 286) shows the solution.

**Additional Resource**

Watch this video if you need additional resource to solve the problem.

<http://y2u.be/UtGtF6nc35g>