Algorithm Report

God's Will Solution

Introduction

The deity resolved to bring forth a cosmos comprising parallel realms adorned with cerulean planets. Within this celestial expanse, these blue planets traverse fixed orbits encircling a red giant. The number of potential orbits consistently exceeds that of the planets at any given instance, and the distances from the red giant to these orbits are subject to stochastic variation. Your task is to assist the deity in discerning the optimal manifestation of divine will concerning the positioning of planets to establish equilibrium. The harmonious equilibrium is attained by minimizing the gravitational interaction between the closest pairs of planets. Consequently, your directive is to advocate for a divine will that dictates the maximum minimum distance between the celestial bodies. Example: Consider the above example with multiple configurations for the following inputs Orbits = [1, 2, 4, 7] Planets = 3

The total Will or configurations are 4, numbered as WILL-i In this example, WILL-3 has the minimum distance among the distances between all 3 planets. So, WILL-3 will create a balance. Note: There are no overlapping orbits. The moon cannot share the same orbits.

1 Methodology

The solution was inspired by the LeetCode problem: Aggressive Cows.

1.1 brute-force method

The simple concept of the brute-force method is to check the distances of all possible planets. It finds the difference between the planets using the function called canWePlace. It performs the work by keeping a minimum distance and finds if we can find the no of points specified in the position or not. It works in the following ways:

1. It takes a minimum value of distance like 1,2,3,.. and gets on increasing order.
2. It finds if it can place the planets within the specified distance that we have taken or not.
3. To the maximum value that it reaches to the distance, it keeps the value and returns the possible positions that it gets from the value.

It does this by taking 2 functions. The first function is called bruteForceSolution, which takes an array and checks if we can place all the elements with the given distance or not. For example, if we distance of 1 using the loop of

```
for i in range(1, limit + 1):
```

If we cannot place all the elements in the given positions, it returns the last position.

The second function is canWePlace which looks at orbits,dist, and planets and checks if we can place the elements in the positions that we checked in the array or not. For example: if we have a distance of 4, and we have an array of 1,5,9. We can place the array and it returns true since the minimum distance for the array is 4. But it was the other array, it returned false. So, the truth value is returned in the brute-force method.

1.2 Efficient Method

and low as The efficient methods solve the problem using the binary search method. It takes two points: high and low. The value of low as

```
low=1
```

and high as

```
high = orbits[n - 1] - orbits[0]
```

It does the work of binary search using the mid-position. It checks on the low value and high value alternatively like the binary search.

- It checks if the mid value(high) can fit all the points in a given possible point.
- If it cannot fit, it changes the value of high to mid and keeps the low as same. It updates low value to mid if it can fit all the points.
- It finds again the mid-value and checks if we can fit all the points with the given distance as arguments within the given points.
- It iterates until it finds the mid value which is the maximum and can fit the problem.
- It returns the value that has the maximum value that we can get after the binary search.

1.3 Comparison

While brute-force methods check for all the possible values to find the distance that is possible or not using the function, the efficient method only checks the mid values using binary search. This comparison reduces the execution time of comparison by totally limiting all possible checks and allowing to fit the planets with a limited number of checks. The search range for all the values differs in both cases.

For the remaining part, we generate 100 random possible orbits and planets taken from the subset of orbits and we use the brute-force and efficient methods of comparison to find the value using the brute-force methods and check the correctness of the output using the efficient methods.

1.4 Output

```
Test Case 1:
Orbits: [40, 57, 59, 74, 95]
K: 4
Brute-Force Result: [40, 57, 74, 95]
Efficient Result: [40, 57, 74, 95]

Test Case 2:
Orbits: [9, 10, 13, 84]
K: 3
Brute-Force Result: [9, 13, 84]
Efficient Result: [9, 13, 84]

Test Case 3:
```

```
Orbits: [14, 15, 36, 44, 56, 63, 84]
K: 5
Brute-Force Result: [14, 36, 44, 56, 84]
Efficient Result: [14, 36, 44, 56, 84]

Test Case 4:
Orbits: [30, 49, 53, 69, 80]
K: 3
Brute-Force Result: [30, 53, 80]
Efficient Result: [30, 53, 80]

Test Case 5:
Orbits: [7, 18, 25, 27, 84]
K: 4
Brute-Force Result: [7, 18, 27, 84]
Efficient Result: [7, 18, 27, 84]

Test Case 6:
Orbits: [30, 34, 49, 68, 70, 81, 82, 86, 96]
K: 6
Brute-Force Result: [30, 49, 68, 81, 86, 96]
Efficient Result: [30, 49, 68, 81, 86, 96]

Test Case 7:
Orbits: [4, 21, 25, 40, 44, 46, 58, 65]
K: 4
Brute-Force Result: [4, 25, 44, 65]
Efficient Result: [4, 25, 44, 65]

Test Case 8:
Orbits: [4, 8, 13, 14, 54, 72, 76, 79, 81]
K: 5
Brute-Force Result: [4, 13, 54, 72, 81]
Efficient Result: [4, 13, 54, 72, 81]

Test Case 9:
Orbits: [14, 17, 29, 64, 94, 96, 97]
K: 6
Brute-Force Result: [14, 17, 29, 64, 94, 97]
Efficient Result: [14, 17, 29, 64, 94, 97]

Test Case 10:
Orbits: [10, 13, 27, 39, 81, 88, 98]
K: 3
Brute-Force Result: [10, 39, 81]
Efficient Result: [10, 39, 81]

Test Case 11:
```

```
Orbits: [5, 21, 44, 68, 70, 78, 80, 84, 99]
K: 5
Brute-Force Result: [5, 21, 44, 68, 84]
Efficient Result: [5, 21, 44, 68, 84]

Test Case 12:
Orbits: [7, 8, 35, 43, 61, 65]
K: 4
Brute-Force Result: [7, 35, 43, 61]
Efficient Result: [7, 35, 43, 61]

Test Case 13:
Orbits: [13, 29, 36, 74, 84]
K: 3
Brute-Force Result: [13, 36, 74]
Efficient Result: [13, 36, 74]

Test Case 14:
Orbits: [47, 49, 62, 91]
K: 3
Brute-Force Result: [47, 62, 91]
Efficient Result: [47, 62, 91]

Test Case 15:
Orbits: [6, 28, 74, 81, 87, 88, 98]
K: 6
Brute-Force Result: [6, 28, 74, 81, 88, 98]
Efficient Result: [6, 28, 74, 81, 88, 98]

Test Case 16:
Orbits: [7, 32, 50, 57, 59, 67, 78, 81, 96, 99]
K: 8
Brute-Force Result: [7, 32, 50, 57, 67, 78, 81, 96]
Efficient Result: [7, 32, 50, 57, 67, 78, 81, 96]

Test Case 17:
Orbits: [15, 32, 54, 58, 67, 71, 74]
K: 4
Brute-Force Result: [15, 32, 54, 71]
Efficient Result: [15, 32, 54, 71]

Test Case 18:
Orbits: [15, 58, 73, 76, 77]
K: 3
Brute-Force Result: [15, 58, 77]
Efficient Result: [15, 58, 77]

Test Case 19:
```

```
Orbits: [16, 23, 26, 85]
K: 3
Brute-Force Result: [16, 26, 85]
Efficient Result: [16, 26, 85]

Test Case 20:
Orbits: [15, 27, 39, 45, 51, 66, 72, 87]
K: 5
Brute-Force Result: [15, 27, 39, 51, 66]
Efficient Result: [15, 27, 39, 51, 66]

Test Case 21:
Orbits: [9, 13, 48, 82, 84, 97]
K: 3
Brute-Force Result: [9, 48, 97]
Efficient Result: [9, 48, 97]

Test Case 22:
Orbits: [4, 28, 39, 52, 58, 79, 88]
K: 4
Brute-Force Result: [4, 28, 52, 79]
Efficient Result: [4, 28, 52, 79]

Test Case 23:
Orbits: [1, 2, 29, 38, 43, 56, 68, 73, 86, 96]
K: 9
Brute-Force Result: [1, 29, 38, 43, 56, 68, 73, 86, 96]
Efficient Result: [1, 29, 38, 43, 56, 68, 73, 86, 96]

Test Case 24:
Orbits: [4, 6, 24, 25, 31, 74, 76, 78, 82, 84]
K: 6
Brute-Force Result: [4, 24, 31, 74, 78, 82]
Efficient Result: [4, 24, 31, 74, 78, 82]

Test Case 25:
Orbits: [6, 12, 47, 60, 64, 65, 98]
K: 4
Brute-Force Result: [6, 47, 65, 98]
Efficient Result: [6, 47, 65, 98]

Test Case 26:
Orbits: [7, 27, 37, 40, 50, 69, 72, 74, 81]
K: 3
Brute-Force Result: [7, 40, 74]
Efficient Result: [7, 40, 74]

Test Case 27:
```

```
Orbits: [10, 25, 42, 71]
K: 3
Brute-Force Result: [10, 42, 71]
Efficient Result: [10, 42, 71]

Test Case 28:
Orbits: [3, 9, 58, 59, 83]
K: 3
Brute-Force Result: [3, 58, 83]
Efficient Result: [3, 58, 83]

Test Case 29:
Orbits: [1, 35, 54, 57, 84]
K: 4
Brute-Force Result: [1, 35, 57, 84]
Efficient Result: [1, 35, 57, 84]

Test Case 30:
Orbits: [2, 26, 55, 68, 80]
K: 4
Brute-Force Result: [2, 26, 55, 80]
Efficient Result: [2, 26, 55, 80]

Test Case 31:
Orbits: [21, 24, 41, 42, 63, 86]
K: 5
Brute-Force Result: [21, 24, 41, 63, 86]
Efficient Result: [21, 24, 41, 63, 86]

Test Case 32:
Orbits: [12, 27, 70, 90, 95]
K: 3
Brute-Force Result: [12, 70, 95]
Efficient Result: [12, 70, 95]

Test Case 33:
Orbits: [5, 20, 31, 45]
K: 3
Brute-Force Result: [5, 20, 45]
Efficient Result: [5, 20, 45]

Test Case 34:
Orbits: [2, 6, 35, 55, 62, 74, 79, 82, 83, 98]
K: 7
Brute-Force Result: [2, 35, 55, 62, 74, 82, 98]
Efficient Result: [2, 35, 55, 62, 74, 82, 98]

Test Case 35:
```

```
Orbits: [48, 52, 76, 94]
K: 3
Brute-Force Result: [48, 76, 94]
Efficient Result: [48, 76, 94]

Test Case 36:
Orbits: [14, 32, 38, 42, 49, 68, 73, 84]
K: 7
Brute-Force Result: [14, 32, 38, 49, 68, 73, 84]
Efficient Result: [14, 32, 38, 49, 68, 73, 84]

Test Case 37:
Orbits: [7, 26, 67, 81, 96]
K: 4
Brute-Force Result: [7, 26, 67, 96]
Efficient Result: [7, 26, 67, 96]

Test Case 38:
Orbits: [6, 20, 31, 44, 48, 51, 53, 61, 70, 93]
K: 6
Brute-Force Result: [6, 20, 31, 44, 61, 93]
Efficient Result: [6, 20, 31, 44, 61, 93]

Test Case 39:
Orbits: [9, 19, 34, 63]
K: 3
Brute-Force Result: [9, 34, 63]
Efficient Result: [9, 34, 63]

Test Case 40:
Orbits: [16, 27, 40, 52, 69, 74, 77, 79, 84, 97]
K: 8
Brute-Force Result: [16, 27, 40, 52, 69, 77, 84, 97]
Efficient Result: [16, 27, 40, 52, 69, 77, 84, 97]

Test Case 41:
Orbits: [9, 24, 46, 48, 53, 80, 82, 85, 90]
K: 4
Brute-Force Result: [9, 24, 46, 80]
Efficient Result: [9, 24, 46, 80]

Test Case 42:
Orbits: [22, 27, 33, 47, 51, 75, 95]
K: 3
Brute-Force Result: [22, 51, 95]
Efficient Result: [22, 51, 95]

Test Case 43:
```

```
Orbits: [14, 26, 40, 41, 57, 61, 82, 85, 88, 98]
K: 7
Brute-Force Result: [14, 26, 40, 57, 82, 88, 98]
Efficient Result: [14, 26, 40, 57, 82, 88, 98]

Test Case 44:
Orbits: [6, 22, 43, 52]
K: 3
Brute-Force Result: [6, 22, 43]
Efficient Result: [6, 22, 43]

Test Case 45:
Orbits: [5, 28, 40, 56, 69, 96]
K: 5
Brute-Force Result: [5, 28, 56, 69, 96]
Efficient Result: [5, 28, 56, 69, 96]

Test Case 46:
Orbits: [4, 15, 31, 38, 45, 78]
K: 5
Brute-Force Result: [4, 15, 31, 45, 78]
Efficient Result: [4, 15, 31, 45, 78]

Test Case 47:
Orbits: [3, 4, 5, 7, 18, 47, 57, 64]
K: 3
Brute-Force Result: [3, 47, 64]
Efficient Result: [3, 47, 64]

Test Case 48:
Orbits: [19, 26, 27, 71]
K: 3
Brute-Force Result: [19, 27, 71]
Efficient Result: [19, 27, 71]

Test Case 49:
Orbits: [6, 24, 54, 70, 80]
K: 4
Brute-Force Result: [6, 24, 54, 80]
Efficient Result: [6, 24, 54, 80]

Test Case 50:
Orbits: [4, 21, 50, 60, 63, 74, 79]
K: 3
Brute-Force Result: [4, 50, 79]
Efficient Result: [4, 50, 79]

Test Case 51:
```

```
Orbits: [61, 66, 71, 97]
K: 3
Brute-Force Result: [61, 71, 97]
Efficient Result: [61, 71, 97]

Test Case 52:
Orbits: [10, 29, 65, 75, 80, 98, 99]
K: 6
Brute-Force Result: [10, 29, 65, 75, 80, 98]
Efficient Result: [10, 29, 65, 75, 80, 98]

Test Case 53:
Orbits: [6, 33, 48, 53, 62, 83]
K: 4
Brute-Force Result: [6, 33, 62, 83]
Efficient Result: [6, 33, 62, 83]

Test Case 54:
Orbits: [6, 45, 71, 78]
K: 3
Brute-Force Result: [6, 45, 78]
Efficient Result: [6, 45, 78]

Test Case 55:
Orbits: [14, 28, 32, 43, 44, 66, 70, 77]
K: 5
Brute-Force Result: [14, 28, 43, 66, 77]
Efficient Result: [14, 28, 43, 66, 77]

Test Case 56:
Orbits: [7, 9, 10, 37, 39, 49, 72, 77, 81]
K: 3
Brute-Force Result: [7, 39, 72]
Efficient Result: [7, 39, 72]

Test Case 57:
Orbits: [1, 75, 77, 82, 99]
K: 3
Brute-Force Result: [1, 75, 99]
Efficient Result: [1, 75, 99]

Test Case 58:
Orbits: [2, 16, 19, 25, 35, 50, 53, 75, 81, 87]
K: 8
Brute-Force Result: [2, 16, 25, 35, 50, 75, 81, 87]
Efficient Result: [2, 16, 25, 35, 50, 75, 81, 87]

Test Case 59:
```

```
Orbits: [72, 77, 80, 91]
K: 3
Brute-Force Result: [72, 80, 91]
Efficient Result: [72, 80, 91]

Test Case 60:
Orbits: [16, 30, 31, 53, 68, 97, 98]
K: 6
Brute-Force Result: [16, 30, 31, 53, 68, 97]
Efficient Result: [16, 30, 31, 53, 68, 97]

Test Case 61:
Orbits: [1, 10, 20, 33, 52, 76, 81, 82, 94, 99]
K: 8
Brute-Force Result: [1, 10, 20, 33, 52, 76, 82, 94]
Efficient Result: [1, 10, 20, 33, 52, 76, 82, 94]

Test Case 62:
Orbits: [10, 12, 15, 48, 50, 97]
K: 3
Brute-Force Result: [10, 50, 97]
Efficient Result: [10, 50, 97]

Test Case 63:
Orbits: [4, 14, 61, 62]
K: 3
Brute-Force Result: [4, 14, 61]
Efficient Result: [4, 14, 61]

Test Case 64:
Orbits: [8, 21, 28, 40, 66, 86]
K: 5
Brute-Force Result: [8, 21, 40, 66, 86]
Efficient Result: [8, 21, 40, 66, 86]

Test Case 65:
Orbits: [31, 46, 49, 58, 60, 70, 95, 96]
K: 7
Brute-Force Result: [31, 46, 49, 58, 60, 70, 95]
Efficient Result: [31, 46, 49, 58, 60, 70, 95]

Test Case 66:
Orbits: [11, 23, 42, 43, 45, 68, 96, 99]
K: 4
Brute-Force Result: [11, 42, 68, 96]
Efficient Result: [11, 42, 68, 96]

Test Case 67:
```

```
Orbits: [14, 22, 29, 32, 51, 54, 55, 60, 92]
K: 4
Brute-Force Result: [14, 32, 51, 92]
Efficient Result: [14, 32, 51, 92]

Test Case 68:
Orbits: [2, 10, 19, 22, 24, 26, 28, 36, 69]
K: 5
Brute-Force Result: [2, 10, 19, 28, 36]
Efficient Result: [2, 10, 19, 28, 36]

Test Case 69:
Orbits: [16, 49, 52, 56, 57, 87, 88]
K: 3
Brute-Force Result: [16, 52, 88]
Efficient Result: [16, 52, 88]

Test Case 70:
Orbits: [17, 70, 94, 98]
K: 3
Brute-Force Result: [17, 70, 98]
Efficient Result: [17, 70, 98]

Test Case 71:
Orbits: [1, 13, 59, 89, 97]
K: 3
Brute-Force Result: [1, 59, 97]
Efficient Result: [1, 59, 97]

Test Case 72:
Orbits: [48, 50, 56, 99]
K: 3
Brute-Force Result: [48, 56, 99]
Efficient Result: [48, 56, 99]

Test Case 73:
Orbits: [3, 7, 9, 57, 67]
K: 4
Brute-Force Result: [3, 9, 57, 67]
Efficient Result: [3, 9, 57, 67]

Test Case 74:
Orbits: [23, 39, 68, 72]
K: 3
Brute-Force Result: [23, 39, 68]
Efficient Result: [23, 39, 68]

Test Case 75:
```

```
Orbits: [15, 24, 26, 37, 47, 61, 71, 82, 87]
K: 6
Brute-Force Result: [15, 26, 37, 47, 61, 71]
Efficient Result: [15, 26, 37, 47, 61, 71]

Test Case 76:
Orbits: [11, 44, 56, 69, 73, 76, 85, 95]
K: 6
Brute-Force Result: [11, 44, 56, 69, 85, 95]
Efficient Result: [11, 44, 56, 69, 85, 95]

Test Case 77:
Orbits: [5, 26, 43, 46, 57, 62, 66, 74, 81]
K: 5
Brute-Force Result: [5, 26, 43, 62, 81]
Efficient Result: [5, 26, 43, 62, 81]

Test Case 78:
Orbits: [1, 2, 10, 42, 51, 53, 73, 90, 95]
K: 4
Brute-Force Result: [1, 42, 73, 95]
Efficient Result: [1, 42, 73, 95]

Test Case 79:
Orbits: [10, 16, 35, 38, 40, 55, 58]
K: 4
Brute-Force Result: [10, 16, 35, 55]
Efficient Result: [10, 16, 35, 55]

Test Case 80:
Orbits: [4, 17, 31, 32, 53, 61, 72, 86, 95]
K: 8
Brute-Force Result: [4, 17, 31, 53, 61, 72, 86, 95]
Efficient Result: [4, 17, 31, 53, 61, 72, 86, 95]

Test Case 81:
Orbits: [6, 13, 28, 48, 53, 57, 77, 89, 90]
K: 5
Brute-Force Result: [6, 28, 48, 77, 90]
Efficient Result: [6, 28, 48, 77, 90]

Test Case 82:
Orbits: [1, 23, 24, 34, 36, 39, 50, 62, 65, 91]
K: 7
Brute-Force Result: [1, 23, 34, 39, 50, 62, 91]
Efficient Result: [1, 23, 34, 39, 50, 62, 91]

Test Case 83:
```

```
Orbits: [13, 25, 29, 33, 58, 71, 93]
K: 5
Brute-Force Result: [13, 29, 58, 71, 93]
Efficient Result: [13, 29, 58, 71, 93]

Test Case 84:
Orbits: [5, 12, 21, 27, 84, 89, 92]
K: 5
Brute-Force Result: [5, 12, 21, 84, 92]
Efficient Result: [5, 12, 21, 84, 92]

Test Case 85:
Orbits: [1, 18, 25, 42, 43, 55, 56, 70, 75, 89]
K: 8
Brute-Force Result: [1, 18, 25, 42, 55, 70, 75, 89]
Efficient Result: [1, 18, 25, 42, 55, 70, 75, 89]

Test Case 86:
Orbits: [15, 19, 26, 48, 56, 57, 65, 77, 82, 89]
K: 9
Brute-Force Result: [15, 19, 26, 48, 56, 65, 77, 82, 89]
Efficient Result: [15, 19, 26, 48, 56, 65, 77, 82, 89]

Test Case 87:
Orbits: [12, 13, 23, 29, 36, 47, 85]
K: 3
Brute-Force Result: [12, 47, 85]
Efficient Result: [12, 47, 85]

Test Case 88:
Orbits: [25, 31, 66, 71, 80, 84, 94]
K: 4
Brute-Force Result: [25, 66, 80, 94]
Efficient Result: [25, 66, 80, 94]

Test Case 89:
Orbits: [4, 12, 16, 18, 31, 33, 58, 86, 88, 90]
K: 6
Brute-Force Result: [4, 12, 18, 31, 58, 86]
Efficient Result: [4, 12, 18, 31, 58, 86]

Test Case 90:
Orbits: [23, 32, 44, 49, 56, 83]
K: 4
Brute-Force Result: [23, 44, 56, 83]
Efficient Result: [23, 44, 56, 83]

Test Case 91:
```

```
Orbits: [23, 44, 49, 54, 79]
K: 3
Brute-Force Result: [23, 49, 79]
Efficient Result: [23, 49, 79]

Test Case 92:
Orbits: [3, 14, 19, 46, 65, 70, 74, 75, 79, 98]
K: 7
Brute-Force Result: [3, 14, 19, 46, 65, 70, 75]
Efficient Result: [3, 14, 19, 46, 65, 70, 75]

Test Case 93:
Orbits: [1, 5, 32, 48, 54, 62, 72, 77, 93, 99]
K: 5
Brute-Force Result: [1, 32, 54, 77, 99]
Efficient Result: [1, 32, 54, 77, 99]

Test Case 94:
Orbits: [9, 18, 21, 73, 77, 84, 88, 89, 95]
K: 6
Brute-Force Result: [9, 18, 73, 84, 89, 95]
Efficient Result: [9, 18, 73, 84, 89, 95]

Test Case 95:
Orbits: [12, 15, 71, 75, 92]
K: 3
Brute-Force Result: [12, 71, 92]
Efficient Result: [12, 71, 92]

Test Case 96:
Orbits: [23, 31, 45, 48, 64, 67, 76, 79, 86, 97]
K: 9
Brute-Force Result: [23, 31, 45, 48, 64, 67, 76, 79, 86]
Efficient Result: [23, 31, 45, 48, 64, 67, 76, 79, 86]

Test Case 97:
Orbits: [36, 43, 48, 87]
K: 3
Brute-Force Result: [36, 48, 87]
Efficient Result: [36, 48, 87]

Test Case 98:
Orbits: [17, 34, 49, 56, 62, 67, 90, 99]
K: 3
Brute-Force Result: [17, 56, 99]
Efficient Result: [17, 56, 99]

Test Case 99:
```

```
Orbits: [23, 34, 47, 73, 84]
K: 4
Brute-Force Result: [23, 34, 47, 73]
Efficient Result: [23, 34, 47, 73]

Test Case 100:
Orbits: [10, 27, 41, 55, 62, 93, 94]
K: 6
Brute-Force Result: [10, 27, 41, 55, 62, 93]
Efficient Result: [10, 27, 41, 55, 62, 93]

All tests passed.
```

References