MONASH University
Information Technology

Advanced Programming

FIT3140

Assignment 4

Team 29 – Matthew Ready, Li Cheng

Name	Student ID
Matthew Ready	25121987
Li Cheng	24864099

Table of contents

Performance measurement

It has been decided that two different approaches to decoding morse code will be investigated, and that the most performant one will be selected. In order to measure performance, we can average the execution time of a program over a number of repetitions. However, we would also like to gain an understanding of how the different methods scale with input size. To answer these questions, we will run 100 repetitions of each method under input sizes of 10,000, 50,000 and 100,000. We will also keep note of the minimum and maximum run times to ensure the stability of the method (i.e. that it doesn't spike at random times).

For the purposes of benchmarking, we have made some simplifications to minimise the amount of code we need to write. First, all input is exactly 4 items (a dash or a dot) in length. Second, only 5 letters are understood, and each of these 5 letters is assumed to consist of 4 items in morse code. This way, we can minimise code complexity and just compare the difference between the two approaches.

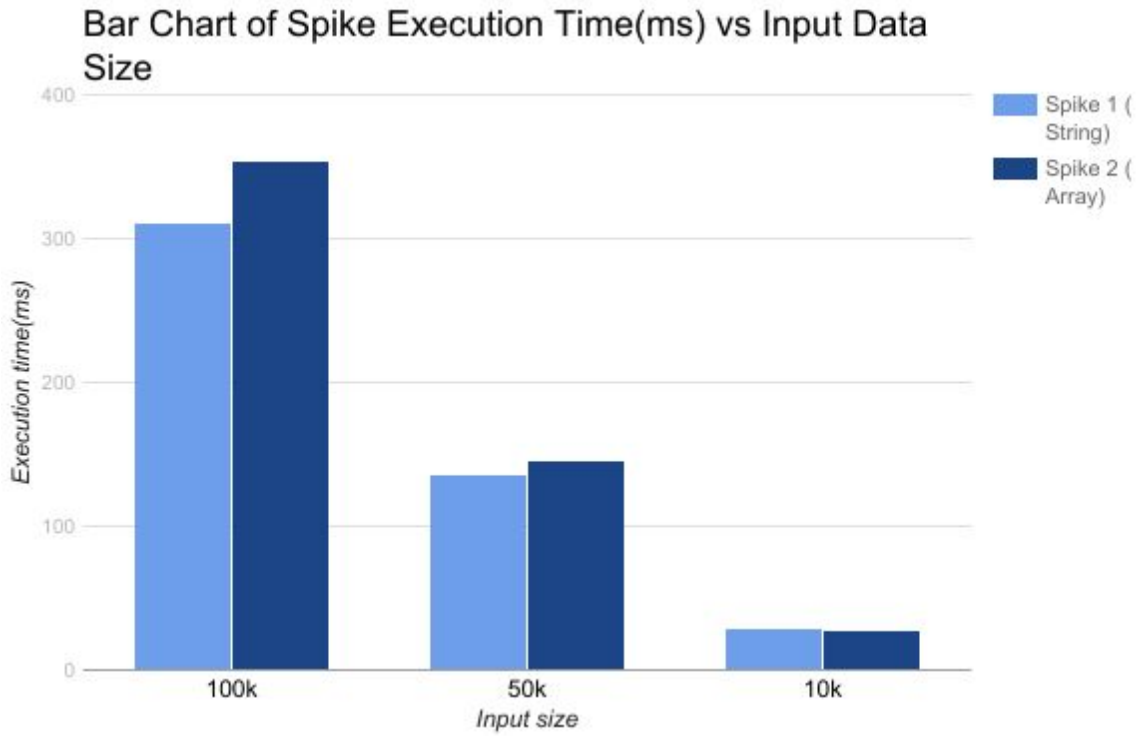
It is also important to note that we have decided to time only the portions of the code responsible for interpreting the morse code. To do this, we first generate the random array and then start a timer using the high performance timer made available to us via `process.hrtime()`. This timer will measure the time taken to interpret and print the entire list. Each of the repetitions generates a new list. This way the effect of random chance in list generation is smoothed out.

The two approaches are as follows:

- Spike 1: Morse code is represented as a string of 1s and 0s for dots and dashes respectively.
- Spike 2: Morse code is represented as an array of '1' and '0' strings for dots and dashes respectively.

Results of benchmarking

1. Bar chart plot of spike execution time(ms) vs Data input size(in thousands)



(Note: a test of 100 repetitions is carried out and averaged to determine times)

2. Execution time span of average, shortest and longest execution times

Average execution time

Data size\Spike(method)	Spike 1 (String)	Spike 2 (Array)
100k	310.63	354.32
50k	135.44	146.07
10k	28.73	28.29

For reference

Shortest execution time

Data size\Spike(method)	Spike 1 (String)	Spike 2 (Array)
100k	235.35	247.85
50k	117.27	119.77
10k	21.25	22.90

Longest execution time

Data size\Spike(method)	Spike 1 (String)	Spike 2 (Array)
100k	1156.43	1179.34
50k	335.53	449.87
10k	51.09	63.69

Note:

- Execution time unit: milliseconds(ms).
- Input data size unit: thousands(k).

Conclusion

We conclude that the string approach to decoding morse code should be used for the next project. It is a more performant approach than the array of characters counterpart. In our experiment, as input data size increases from 10k to 100k, the execution time of the string approach begins to slowly outperform the array of characters approach. In a low data size level of 10k, it is not obvious. As the data size increases to 50k, there is a 7.28% performance improvement in string approach. As the data size increases to 100k, there is a 12.33% performance improvement in string approach. The mathematics used to calculate the performance boost is as following:

$$\frac{longer\ execution\ time - shorter\ execution\ time}{longer\ execution\ time}$$

Take data input size level of 100k as an example, (354.32 ms - 310.63 ms) / 354.32 ms = 12.33%. A 12.33% performance improvement on string approach.

However, there is a strange result in the average execution time of the approaches when the data is 10k in size, where the string approach is indeed slower than the array approach. To make this testing result consistent with our earlier finding, “As input data size increases, the string approach will outperform array of characters approach in determining the appropriate data structure for morse code decoder”, we had a look at the shortest execution time and the longest execution time table. In both tables, it is obvious that string approach outperforms the array of characters approach. Therefore, this instance can be taken as an outlier of our test results.

In conclusion, we have decided to use the string approach (spike 1) for decoding morse code in our upcoming project.