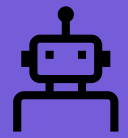
CMP201 – DATA STRUCTURES AND ALGORITHMS 1

BY ISAAC BASQUE-RICE (1901124)

REAL WORLD PROBLEM

PROBLEM

A fictional company, '46AndI', analyses its customers DNA



It uses genome sequencing to let its customers know all kinds of things about themselves

They have asked me to compare two algorithms that search a large amount of text for certain user-defined patterns

PROBLEM

Genetic sequences have a very limited character set (G, A, T, C)

The patterns therefore tend to get quite long, and the text even longer still

SOLUTION

ALGORITHMS

Boyer-Moore

- Best case O(n/m)
- Worst case O(mn)
- Usually considered standard
  - (probably) used in most ctrl+F functions

Rabin-Karp

- Average AND Best-case O(n+m)
- Worst case O(mn)
- Like Boyer-Moore
  - But doesn't immediately consider content, only length and hash value

ABSTRACT DATA TYPES

WHERE THEY DIFFER FROM STANDARD

MACROS

- Used here instead of const int
- Set when program is compiled

```
#define d 256 // macro for ASCII charset
```

ARRAYS

- inPattern is an array of Boolean values, if inPattern[x] == true, ascii x is in pattern
- The program can choose the position it needs to enter or retrieve data from the array
- Time complexity is O(1)

```
bool inPattern[256]{};
```

VECTORS

- Vector used for file open function
- A dynamic array (therefore O(1)), which is useful for storing text files

```
vector<char> buf(length);
```

COMPARISONS

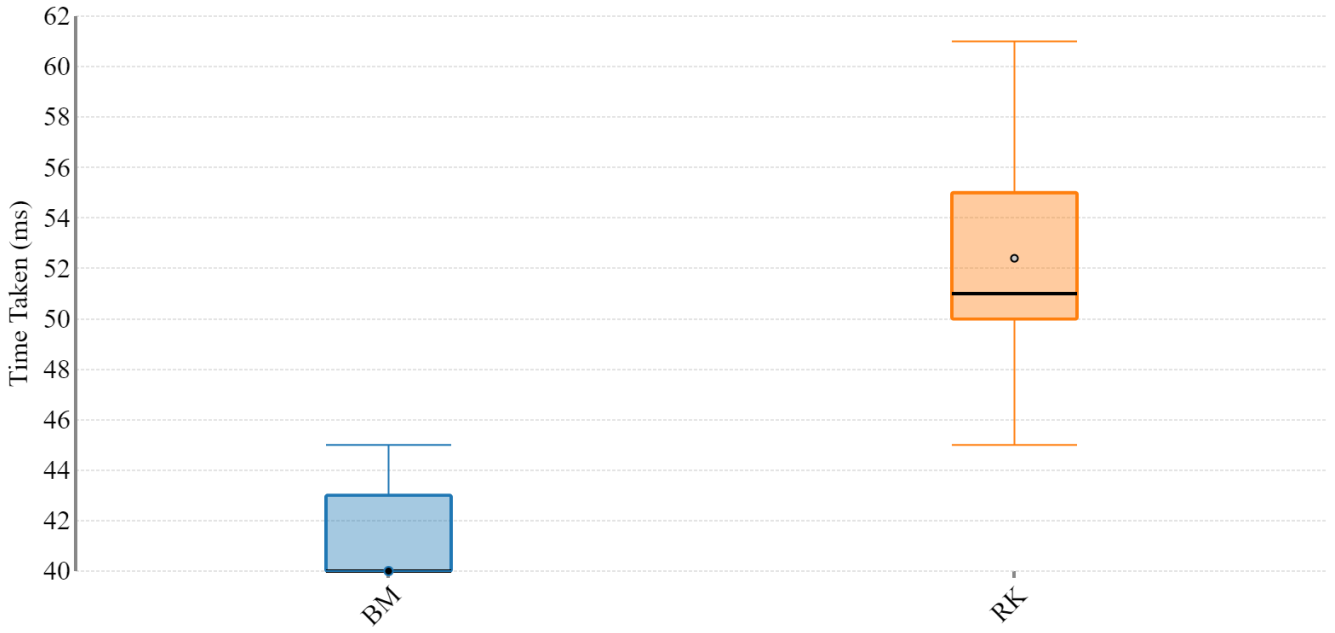
HYPOTHESIS

- As touched on previously, Boyer-Moore best case = O(n/m) and worst case = O(nm)
- Rabin-Karp best and average case = O(n+m) and worst case = O(nm)
- BOTH algorithms likely to perform near worst case in some situations
  - Rabin-Karp better when files larger (not skip reliant)

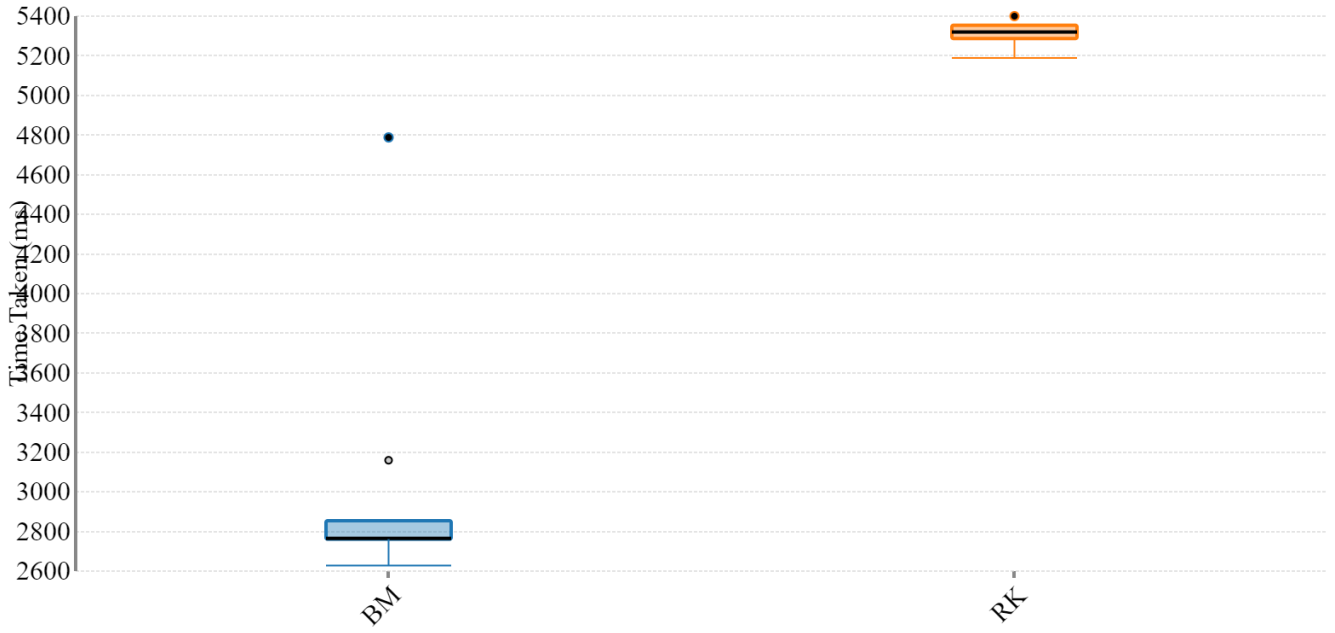
EXPLANATION OF PROCESS

- 3 text files, each 2 orders of magnitude bigger than the last (approximately)
- The next few slides will be of box plots of the amount of time each algorithm took to get through each size file
- The size of the input will be varied, as will the size of the file
- Each test will be conducted 5 times and a box plot will be created from that range
- Each string will be taken at random from the text file in question

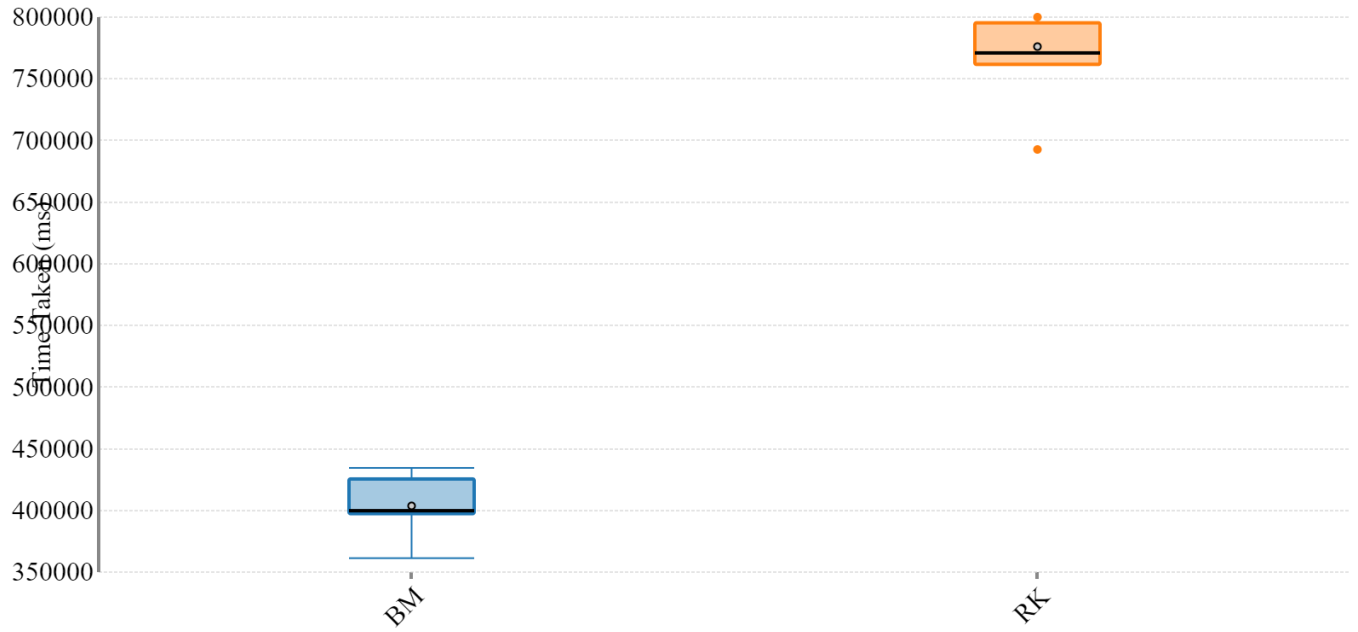
SMALL INPUT/SMALL TEXT



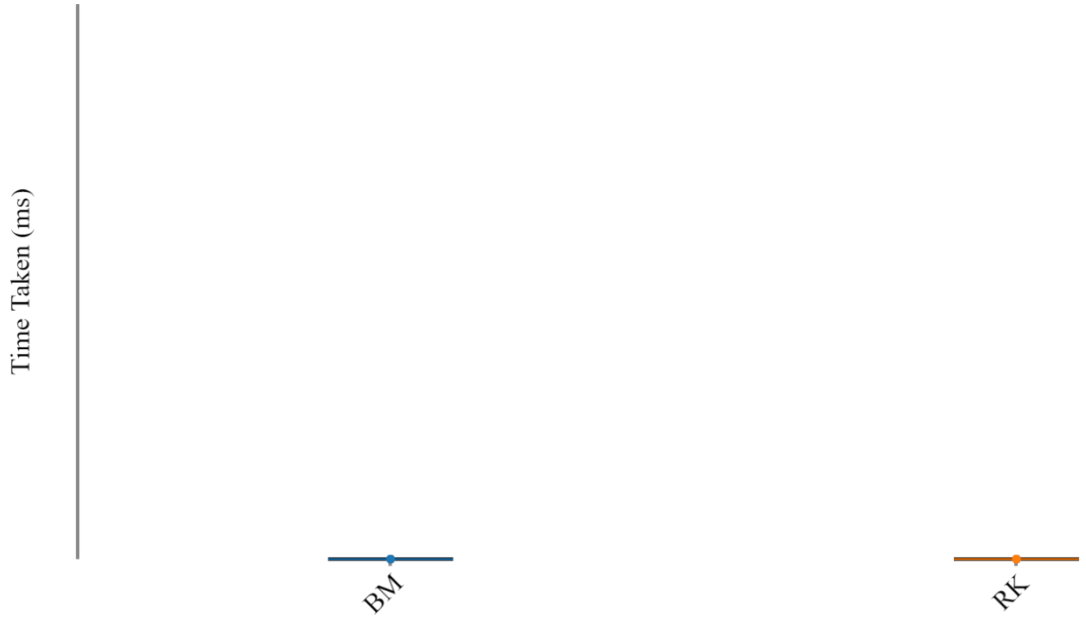
SMALL INPUT/MEDIUM TEXT



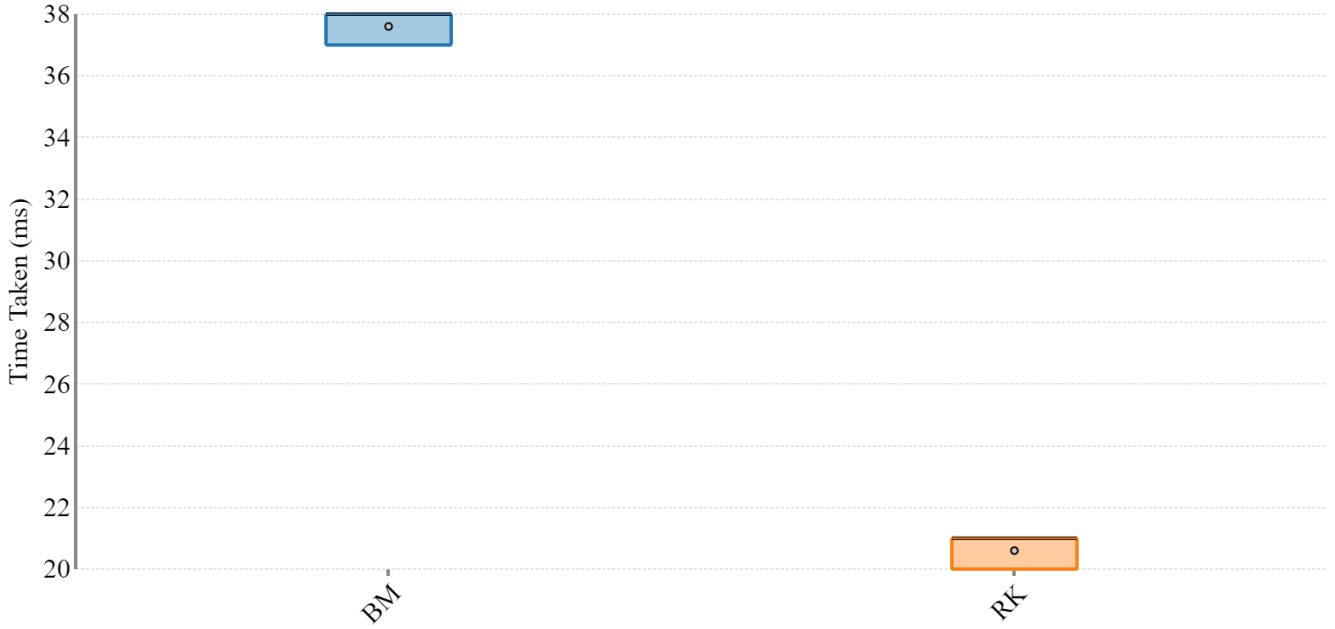
SMALL INPUT/LARGE TEXT



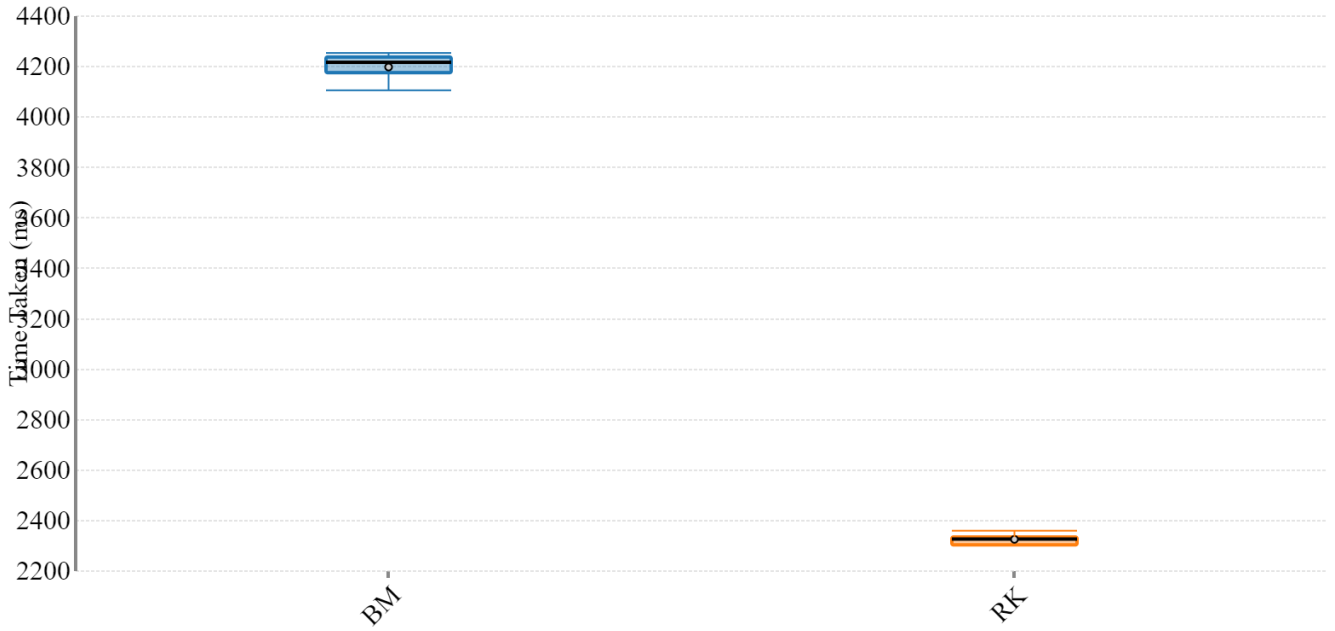
MEDIUM INPUT/SMALL TEXT



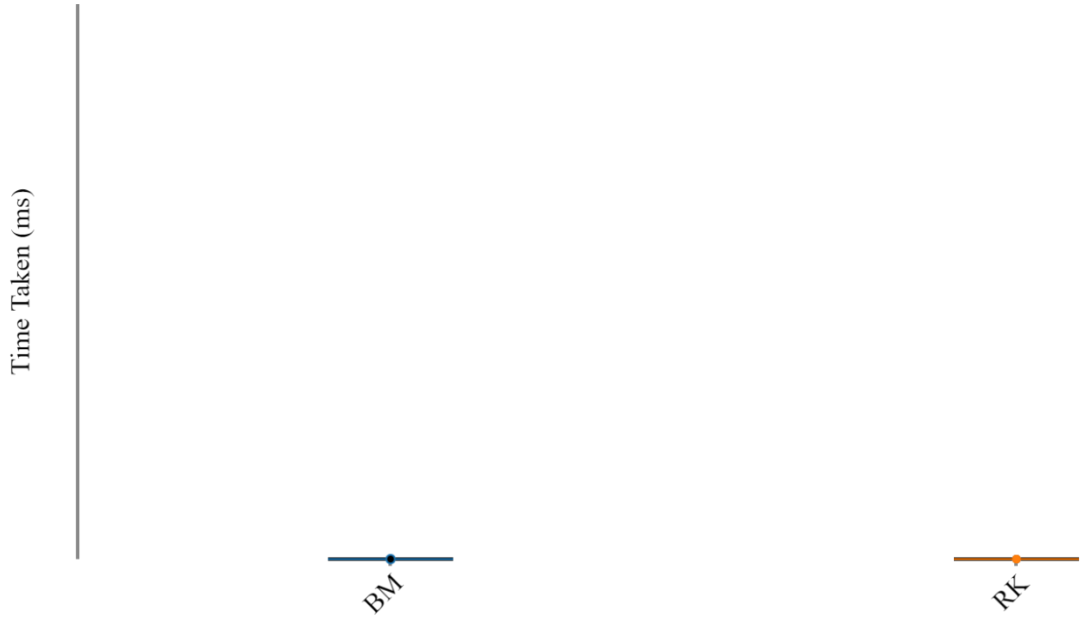
MEDIUM INPUT/MEDIUM TEXT



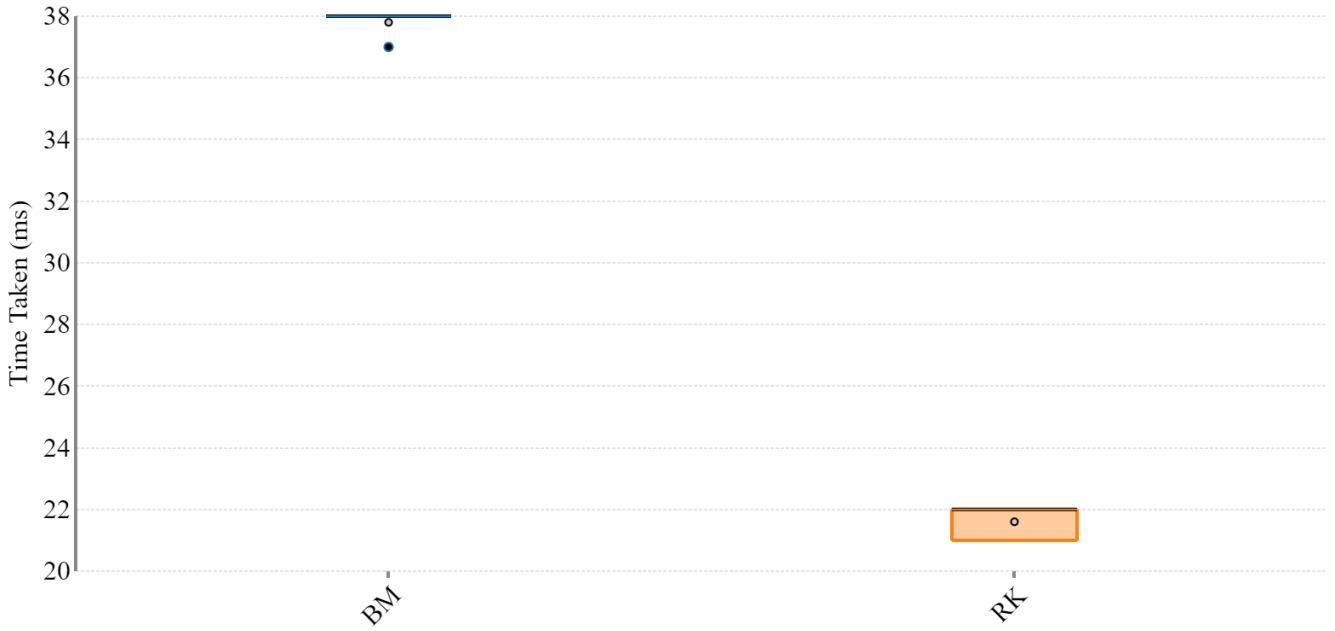
MEDIUM INPUT/LARGE TEXT



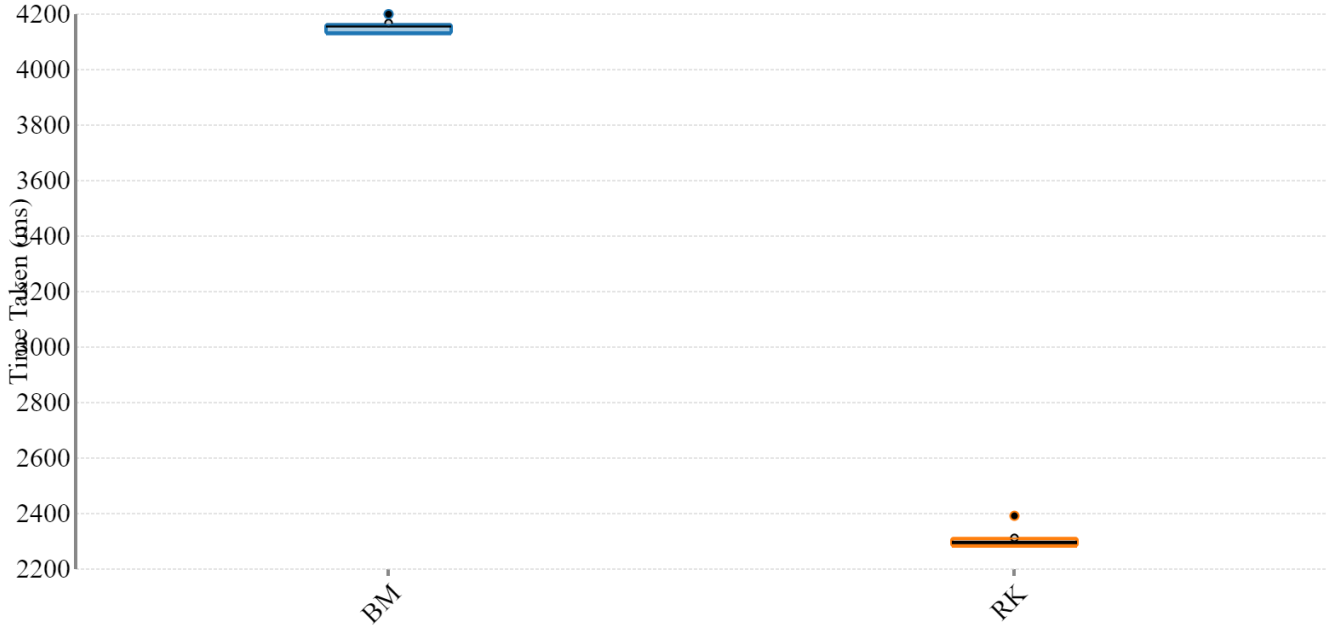
LARGE INPUT/SMALL TEXT



LARGE INPUT/MEDIUM TEXT



LARGE INPUT/LARGE TEXT



POSSIBLE SOURCES OF ERROR

- Developed across multiple platforms (Windows and Manjaro Linux)
- Built in functions measure time (std::chrono::high_resolution_clock)

CONCLUSION

CONCLUSION

- Rabin-Karp improves as the size of the pattern increases/the number of hits decreases
- This is because Boyer-Moore doesn't deal well with smaller charsets
- However if the input is smaller Boyer-Moore performs better
- Out of the two algorithms I would recommend Rabin-Karp here (although I imagine there are better algorithms available)

THANK YOU!