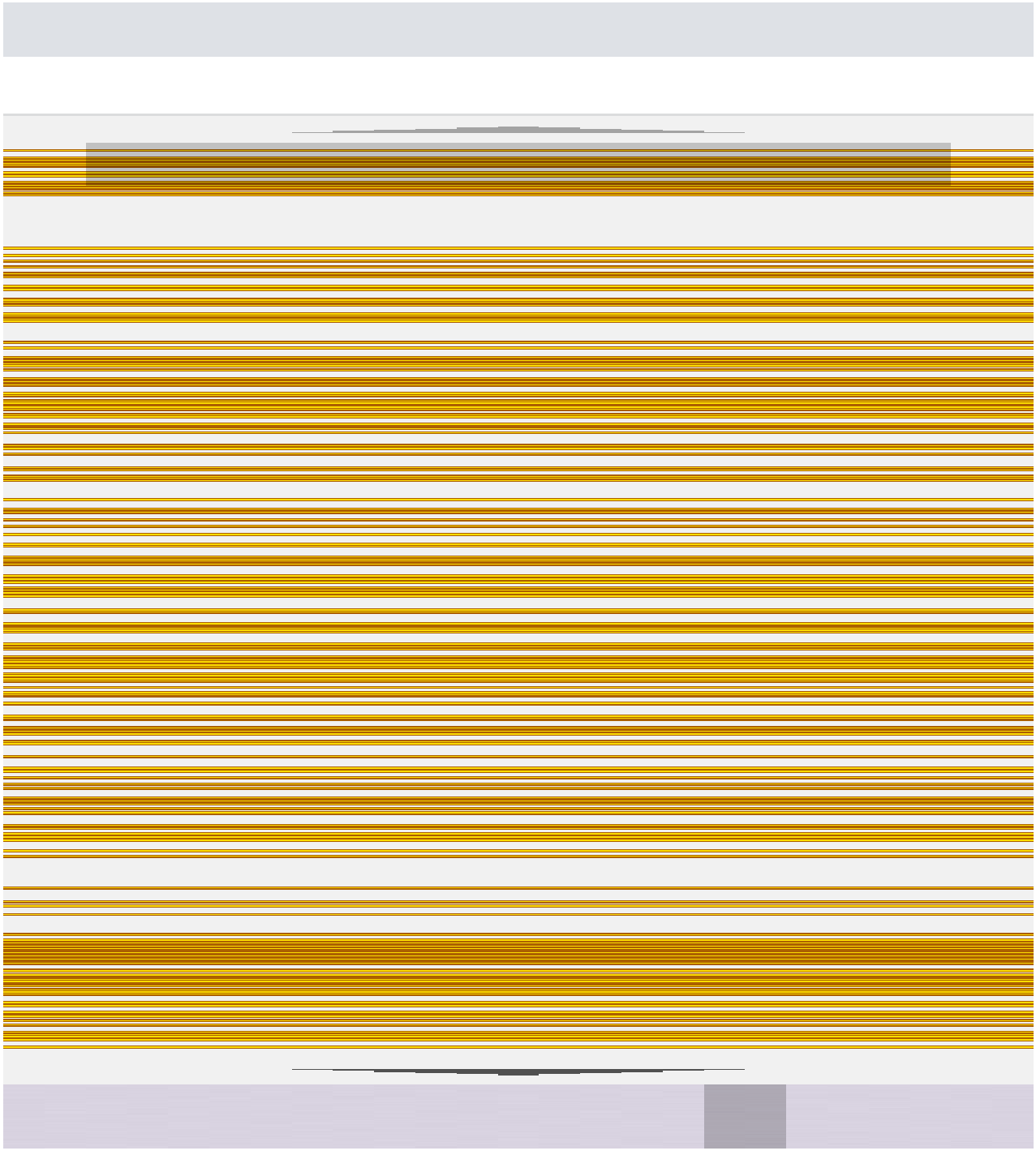
Shandon Fleming

CART 451

Let's Communicate

Data Visualization:



Boyer Moore Pattern Matching Algorithm

-The Boyer Moore Pattern Matching algorithm is used to locate and annotate occurrences of words on a page as well as calculate their frequencies.

How it Works (the process)

Step 1: create an empty map of size 256 (because the maximum number of characters that can be represented in extended ASCII is 256) and set to -1

```
let string = "ATAATTACCAACATCATAATTACCAACATCATAATTACCAACATCATAATTACCAA
let pattern = "ATC";

let M = pattern.length;
let N = string.length;
let skip;                           //to determine substring skip
let res = [];                       //to store result

let map = new Array(256);           //array of 256 length
```

Step 2: map the character(s) to its index in the pattern

```
for(let c = 0;c<256;c++){
  map[c] = -1;                      //initialize to -1
}

for(let j=0;j<M;j++){
  map[pattern[j]] = j;              //initialize to the it's index
}
```

Step 3: Loop over the string. In the for loop, instead of "i++", use i+= skip, to skip that part of the string.

```
for(let i=0;i<=N-M;i+=skip)
```

Step 4: Set skip to 0 during each iteration (important)

```
for(let i=0;i<=N-M;i+=skip){
  skip=0;
}
```

Step 5: Match the pattern with it's relevant string

```
for(let i=0;i<=N-M;i+=skip){
  skip=0;
  for(let j = M-1;j>=0;j--){

    if(pattern[j] != string[i+j]){
      skip = Math.max(1,j-map[string[i+j].charCodeAt(0)]);
      break;
    }
  }
}
```

Step 6: If there is a mismatch, find the length that must be skipped and perform a skip like

```
    skip = Math.max(1,j-map[string[i+j]]);
```

Note:

In instances like "ACC" and "ATC", the last character's match but rest do not. Instinctively, one would want go back and match the first "C" of the string with "C" of pattern, but doing so will imply that one is going back which should be avoided or else you will be stuck in an infinite loop going back and forth. To ensure that you keep progressing through the matching process, ensure that whenever you come across situations when there's a negative skip, and set skip to 1.

Step 7: If the skip is 0 (if there we no mismatch) add "i" to the result list.

```
if(skip == 0){
    console.log(skip)
    res.push(i);
    skip++;
  }
```

All the steps combined result in:

```
let string = "ATAATTACCAACATCATAATTACCAACATCATAATTACCAACATCATAATTACCAAC
let pattern = "ATC";

let M = pattern.length;
let N = string.length;
let skip;
let res = [];

let map = new Array(256);

for(let c = 0;c<256;c++){
  map[c] = -1;
}

for(let j=0;j<M;j++){
  map[pattern[j]] = j;
}

for(let i=0;i<=N-M;i+=skip){
  skip=0;
  for(let j = M-1;j>=0;j--){

    if(pattern[j] != string[i+j]){
      skip = Math.max(1,j-map[string[i+j].charCodeAt(0)]));
      break;
    }
  }
  if(skip == 0){
    res.push(i);
    skip++;
  }
}

console.log(res);
```

How you can use the Boyer Moore Pattern Matching Algorithm in your everyday life

-You can use the Boyer Moore Pattern Matching Algorithm by pressing ctrl-f

Real world example: