# Idea 1 buckets of words

What would the working file look like?

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Word | ‘005’ | ‘014’ | ‘104’ | ‘113’ |  | ‘320’ | ‘401’ | ’410’ | ‘500’ |
| Aahed | LOW | LOW | LOW | LOW |  | LOW | LOW | LOW | Aahed |
| Aalii | LOW | LOW | LOW | LOW |  | LOW | LOW | LOW | Aalii |
| Aargh | LOW | LOW | LOW | LOW |  | LOW | LOW | LOW | Aargh |
|  |  |  |  |  |  |  |  |  |  |
| Zygon | LOW | LOW | LOW | LOW |  | LOW | LOW | LOW | Zygon |
| Zymes | LOW | LOW | LOW | LOW |  | LOW | LOW | LOW | Zymes |
| Zymic | LOW | LOW | LOW | LOW |  | LOW | LOW | LOW | Zymic |

Where LOW is a list of words that when compared to the word in the column ‘Word’ (which will be the index) matches the pattern in that column

Algorithm1 given guess and green, yellow, red:

1. Put all words that don’t match these patterns in a list
2. Calculate how many words are still left
3. Going through each index:
4. Set Entropy to 0
5. Going through each column:
6. | Going through each word:
7. | | Delete word if in list from step 1
8. | Entropy = Entropy + (amount of words in field / value step 2)
9. Store combination Word & Entropy
10. Give top x best guesses by entropy

Notes:

* Time:
* All words can still be used as guesses, but as we get close to the solution, we need a better method than entropy to get the solution

Algorithm2 given guess and green, yellow, red:

1. Put all words that don’t match these patterns in a list
2. Calculate how many words are still left
3. Going through each index:
4. If index in list step 1:
5. | Delete row
6. Going through each index:
7. Set Entropy to 0
8. Going through each column:
9. | Going through each word:
10. | | Delete word if in list from step 1
11. | Entropy = Entropy + (amount of words in field / value step 2)
12. store combination Word & Entropy
13. Give top x best guesses by entropy

Notes:

* Time: , where p is the percentage of words that are still a possible solution
* Can get into a situation where you need 3+ guesses because you cannot use a filler word

# Idea 2 n,n matrix with words

What would the working file look like?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Word | Aahed | Aalii | Aargh |  | Zygon | Zymes | Zymic |
| Aahed | ‘500’ | Pattern | Pattern |  | Pattern | Pattern | Pattern |
| Aalii | Pattern | ‘500’ | Pattern |  | Pattern | Pattern | Pattern |
| Aargh | Pattern | Pattern | ‘500’ |  | Pattern | Pattern | Pattern |
|  |  |  |  |  |  |  |  |
| Zygon | Pattern | Pattern | Pattern |  | ‘500’ | Pattern | Pattern |
| Zymes | Pattern | Pattern | Pattern |  | Pattern | ‘500’ | Pattern |
| Zymic | Pattern | Pattern | Pattern |  | Pattern | Pattern | ‘500’ |

Algorithm 3 with guess and green, yellow, red known:

1. For every column in row[guess]:
2. If pattern doesn’t match green, yellow, red:
3. | delete row[column name]
4. | delete column[column name]
5. For every row:
6. Create dict with every pattern with value 0
7. For every column:
8. | dict[pattern] += 1
9. | Entropy = Entropy + (amount of words in field / amount of rows)
10. store combination Word & Entropy
11. Give top x best guesses by entropy

Notes:

* Time:
* Can get into a situation where you need 3+ guesses because you cannot use a filler word