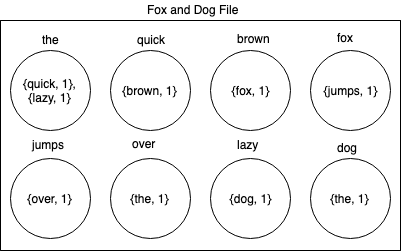
**1-Order System Proof**

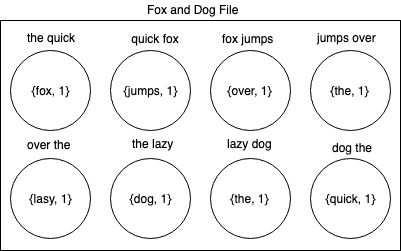
To prove the 1-order system, I will modify the way we think of the adjacency struct. Each object in the struct will be treated as a set, where the object signature is the set’s name and the elements of the set are tuples containing adjacent signatures and their corresponding frequencies. Again, we will use “The quick brown fox jumps over the lazy dog” as sample data for illustration. One thing to note is that the last element in the set is assigned an adjacency to the first element. This ensures at least one adjacency with the last element of the data set. In this example, if dog was not adjacent to the there would be no adjacencies for dog.



Because the system can only select from a set’s adjacent signatures, it is guaranteed that there will be a signature to traverse to at any time. This is pretty clear in a 1-order system.

**N-Order System Proof**

**­**In an n-order system the complexity comes in making sure what has been written in the song thus far can map to a sequence of signatures.



One thing to note is that the last element loops back to the beginning of the file; having its signature comprised of the last and first tokens. This is necessary to ensure a path is the program reaches this token. As the order system increases, so does this overlap.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 0 |
| 1 | the | quick |  |  |  |  |  |  |  |
| 2 |  | quick | fox |  |  |  |  |  |  |
| 3 |  |  | fox | jumps |  |  |  |  |  |
| 4 |  |  |  | jumps | over |  |  |  |  |
| 5 |  |  |  |  | over | the |  |  |  |
| 6 |  |  |  |  |  | the | lazy |  |  |
| 7 |  |  |  |  |  |  | lazy | dog |  |
| 8 |  |  |  |  |  |  |  | dog | the |
| num | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

In this graph every row is a signature sequence. Above every token is its index in the list. In the last signature sequence, index 7 and 0 are used to construct ‘dog the.’

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 0 | 1 |
| 1 | the | quick | fox |  |  |  |  |  |  |  |
| 2 |  | quick | fox | jumps |  |  |  |  |  |  |
| 3 |  |  | fox | jumps | over |  |  |  |  |  |
| 4 |  |  |  | jumps | over | the |  |  |  |  |
| 5 |  |  |  |  | over | the | lazy |  |  |  |
| 6 |  |  |  |  |  | the | lazy | dog |  |  |
| 7 |  |  |  |  |  |  | lazy | dog | the |  |
| 8 |  |  |  |  |  |  |  | dog | the | quick |
| num | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

In this graph there is a similar pattern to the first with its overlap. In this one though, the overlap spans the first two indices of the list. It is also true that if you have an order-four system the overlap spans the first three indices and so on.