

Aho-Corasick Algorithm – Multiple Pattern matching

ChaoHui Zheng
University of Tennessee
CS594 Class
October 5 2021

Problem

- Given Patterns P1, P2, P2, P3,
- Given a text string T
- Find all occurrences of P1, P2, P3 ... in text T

Applications

- Regular expression
- Plagiarism detection
- Spell checking
- Popular Interview question

Input/Output example

Pattens

A		
C		
A	B	
B	C	
B	A	B
B	C	A
C	A	A

Diagram illustrating the execution of a query plan on a table with columns A, B, C, C, A, B. The plan consists of a join of two projections. The first projection selects columns A, B, and C. The second projection selects columns C, A, and B. The join is performed on the column C of the first projection and the column C of the second projection. The diagram shows the execution of the join operation, with the first projection reading rows from the table and the second projection reading rows from the table. The join operation is shown as a red line connecting the two projections.

Naïve approach

- For every index i in text T , we check all patterns
- Big O: $O(L(T) * (L(P1) + L(P2) + L(P3) \dots))$

Process Index 0

A	B	C	C	A	B
A					
C					
A	B				
B	C				
B	A	B			
B	C	A			
C	A	A			

Naïve approach

- For every index i in text T , we check all patterns
- Big O: $O(L(T) * (L(P1) + L(P2) + L(P3) \dots))$

Process Index 1

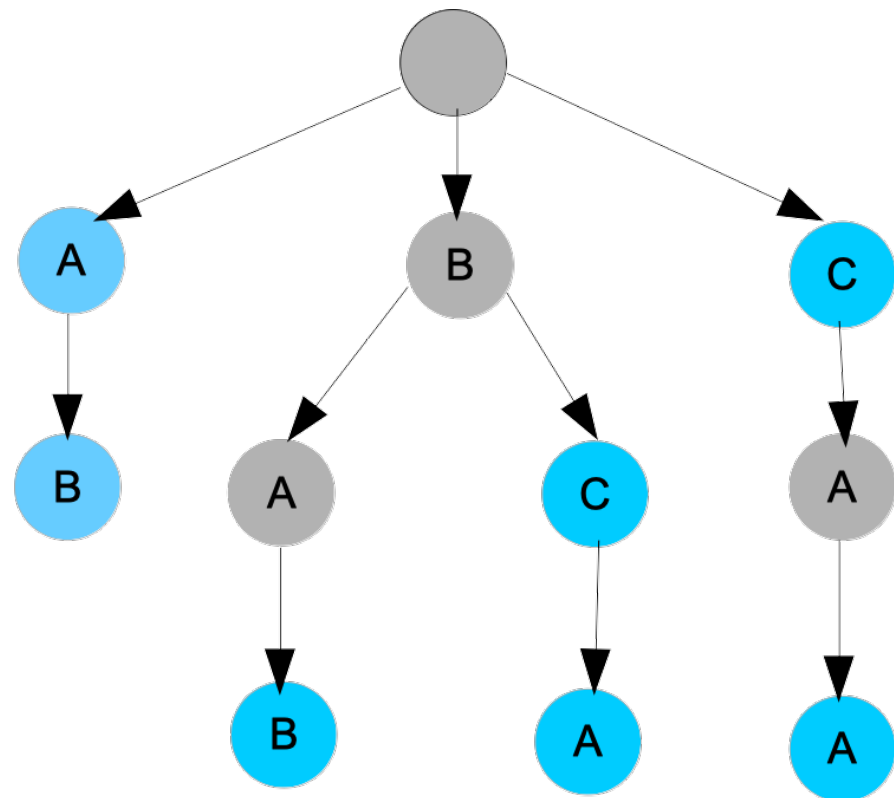
A	B	C	C	A	B
	A				
	C				
	A	B			
	B	C			
	B	A	B		
	B	C	A		
	C	A	A		

Can we do better?

Aho-Corasick

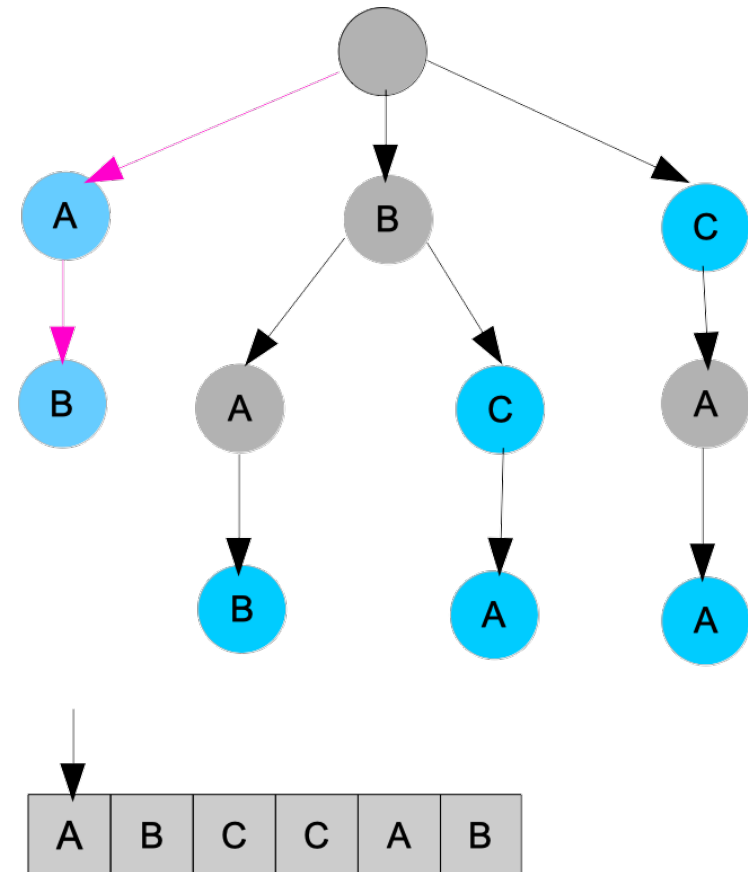
1. Build the graph representation for patterns (Trie)

A		
C		
A	B	
B	C	
B	A	B
B	C	A
C	A	A



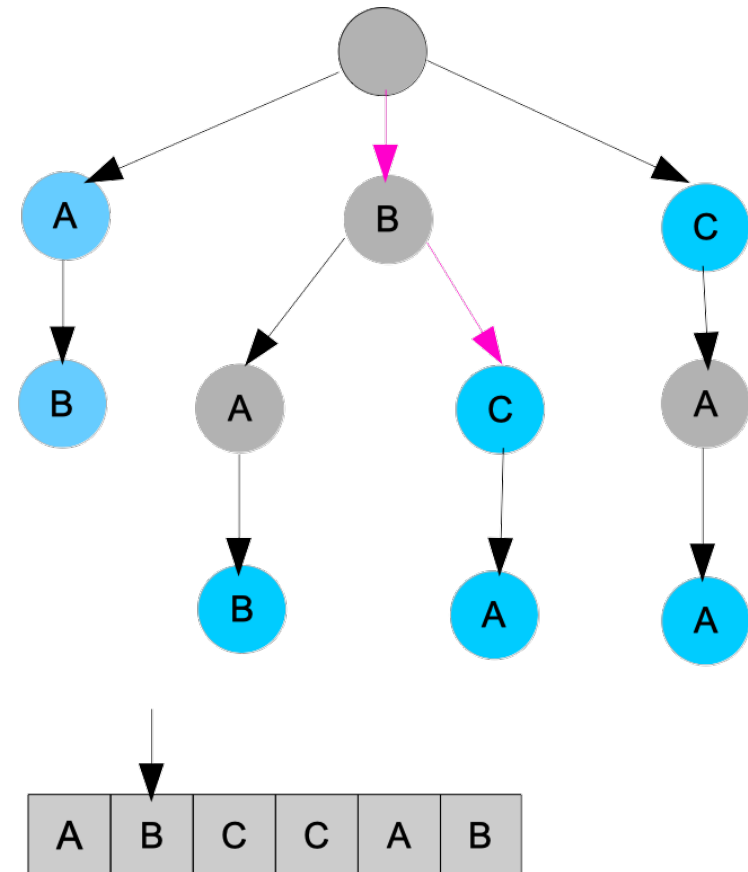
Aho-Corasick

1. Build the graph representation for patterns (Trie)



Aho-Corasick

1. Build the graph representation for patterns (Trie)

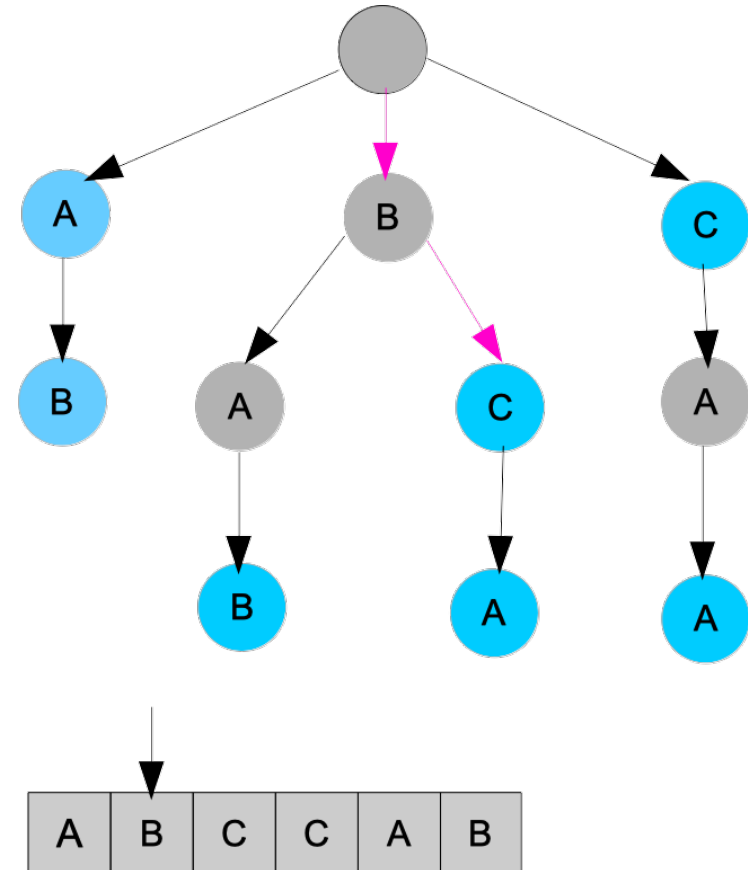


Aho-Corasick

1. Build the graph representation for patterns (Trie)

$O(L(T) * L_{\max}(P1, P2, P3, \dots))$

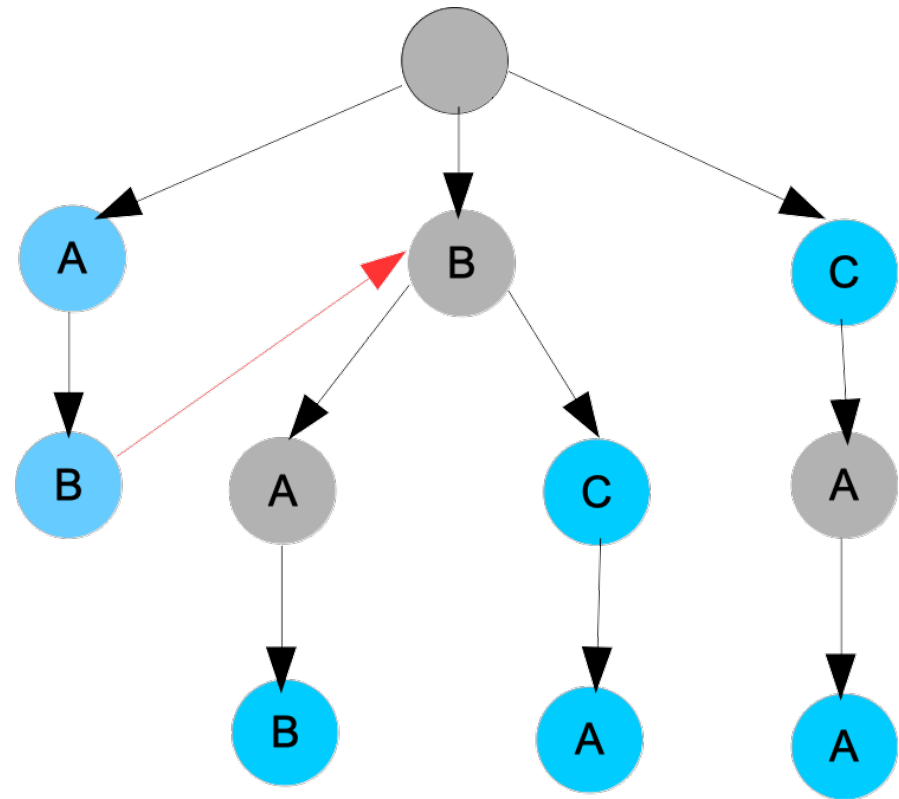
Can we do better?



Aho-Corasick

1. Build the Trie
2. Add failure links.

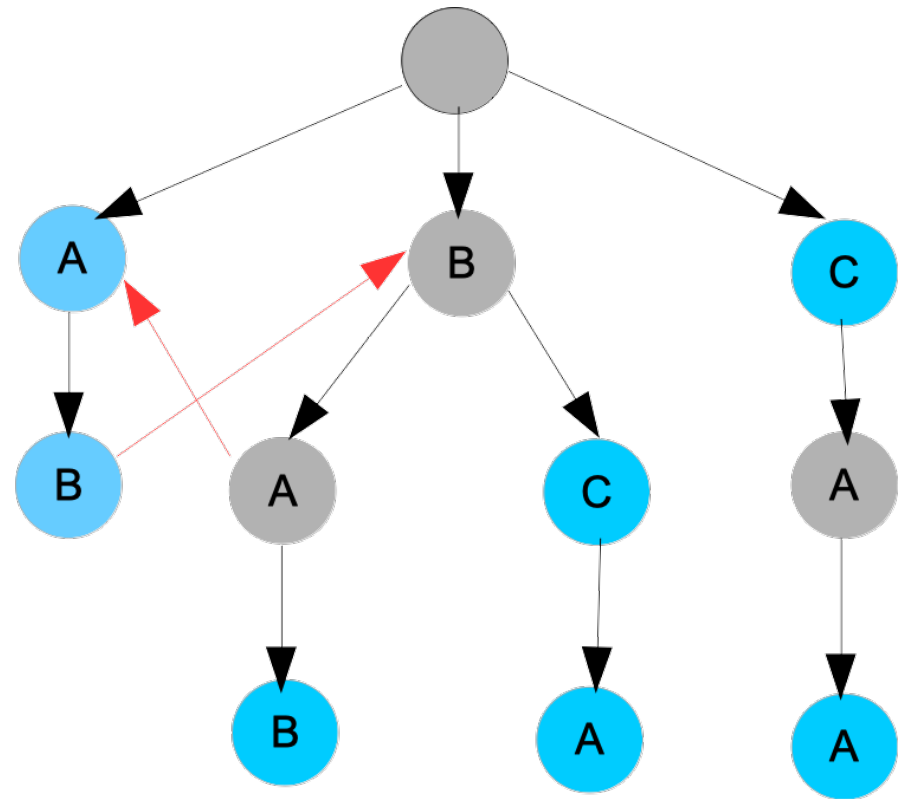
Path AB
Path BAB
Max overlap B



Aho-Corasick

1. Build the Trie
2. Add failure links.

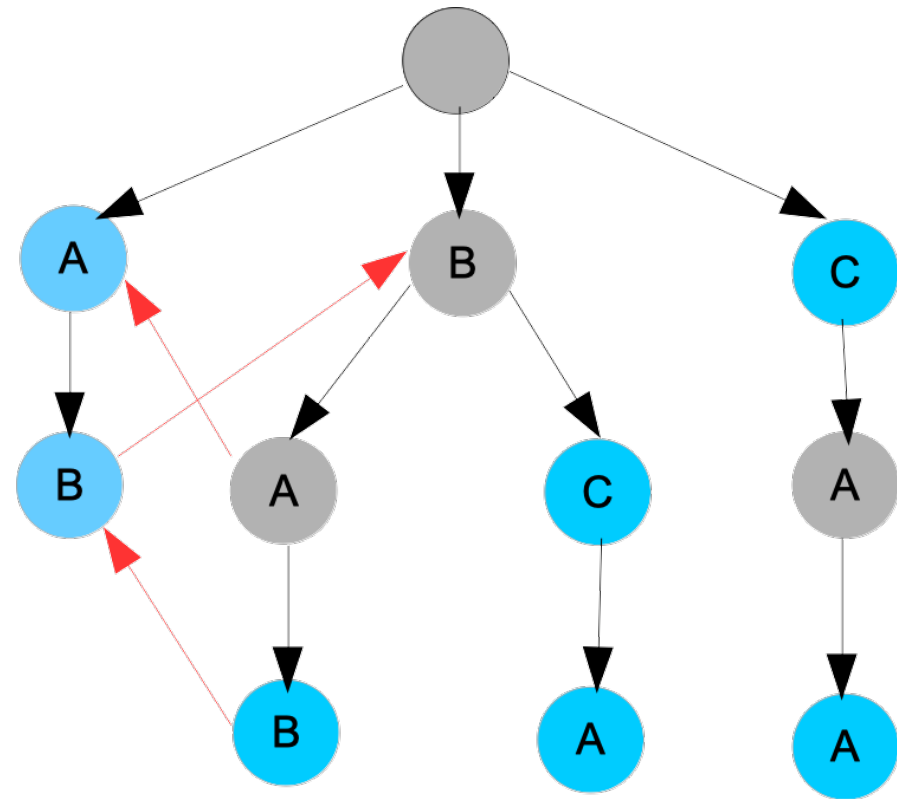
Path BA
Path AB
Max Overlap A



Aho-Corasick

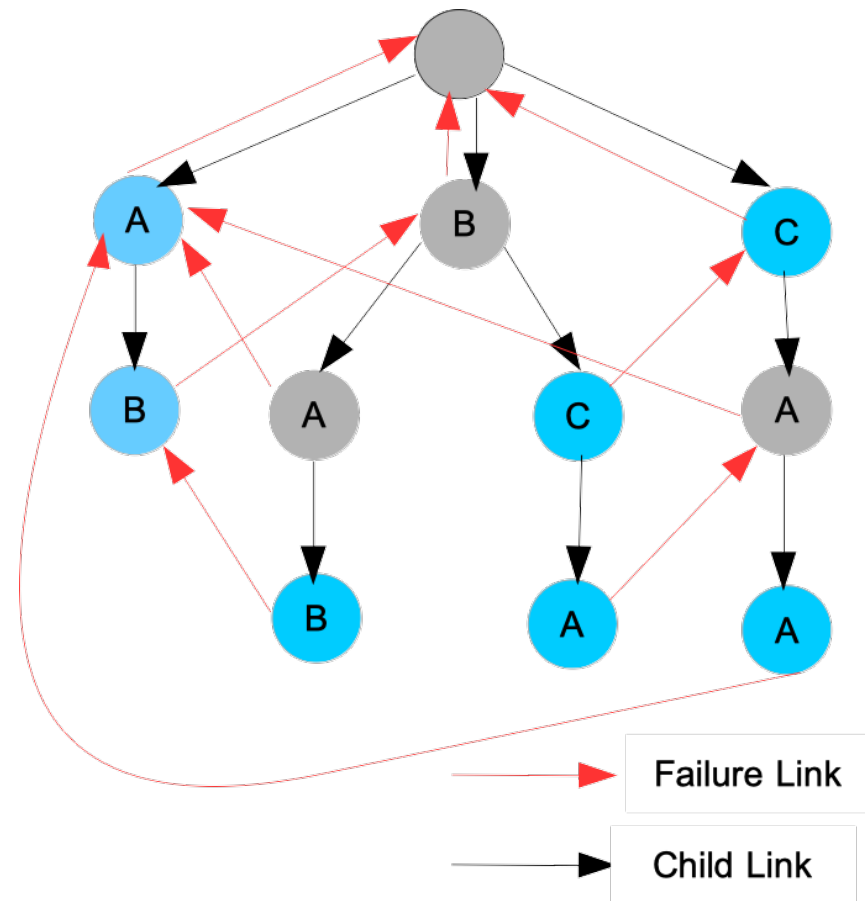
1. Build the Trie
2. Add failure links.

Path BAB
Path AB/BCA
Max Overlap AB



Aho-Corasick

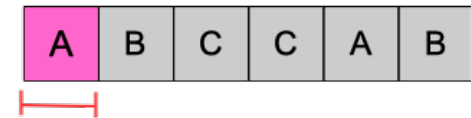
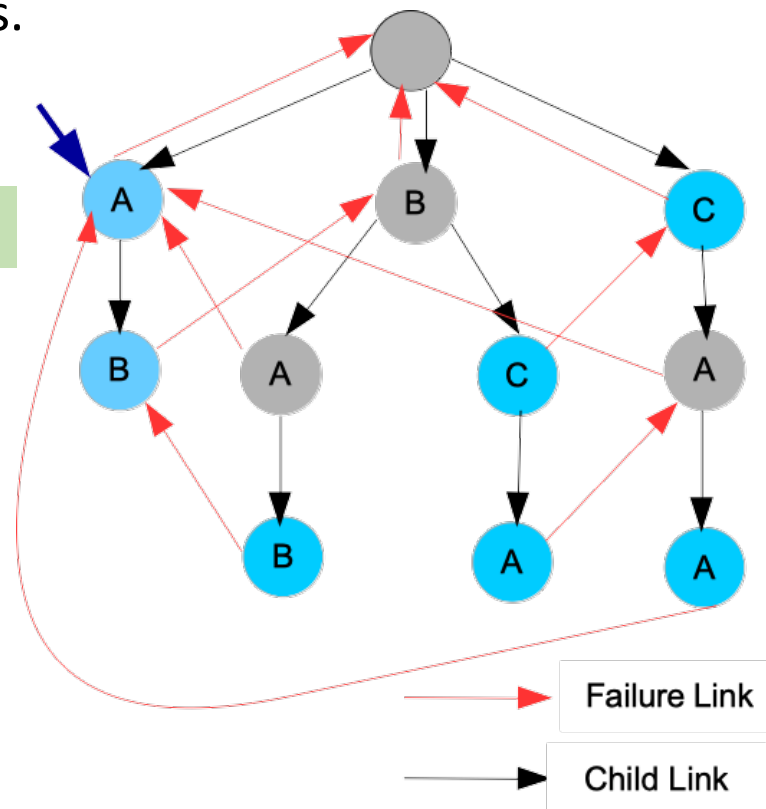
1. Build the Trie
2. Add failure links.



Aho-Corasick

1. Build the Trie
2. Add failure links.

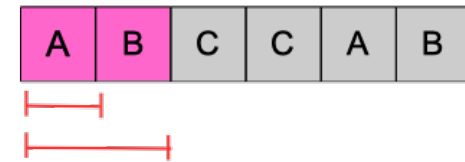
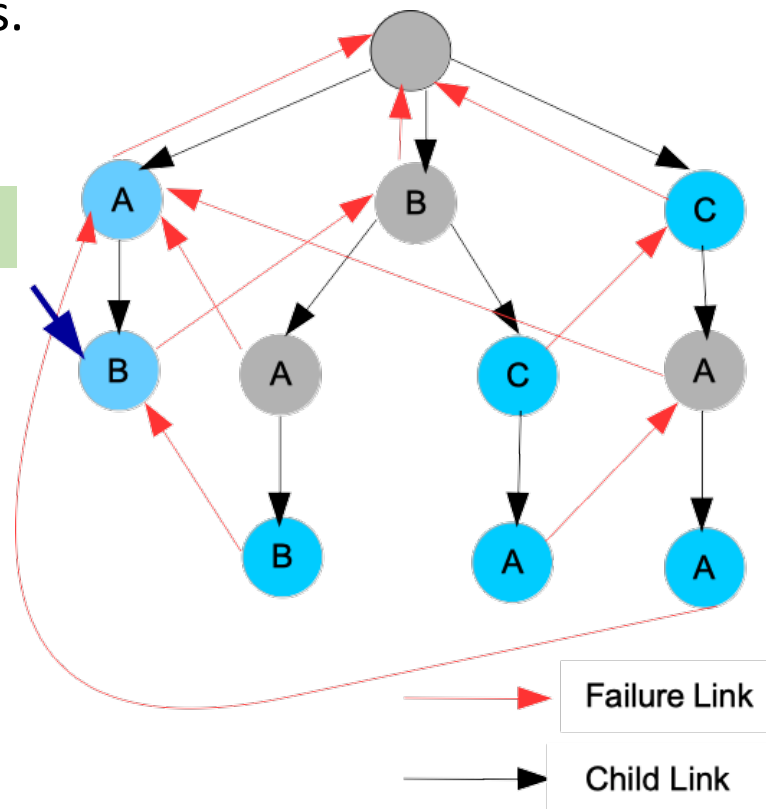
Match A



Aho-Corasick

1. Build the Trie
2. Add failure links.

Match AB

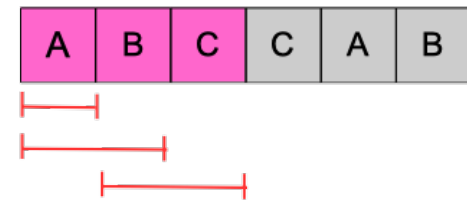
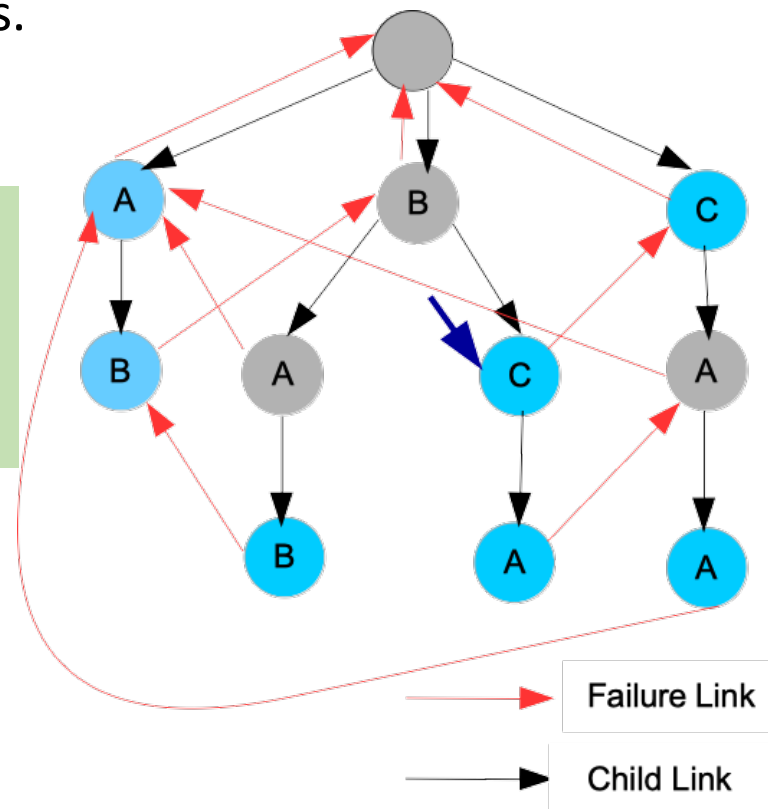


Aho-Corasick

1. Build the Trie
2. Add failure links.

Follow failure link
to middle B

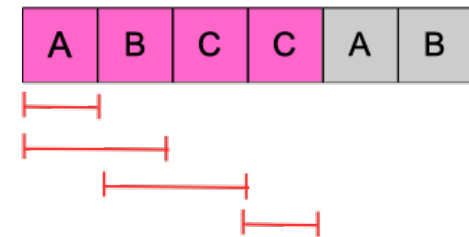
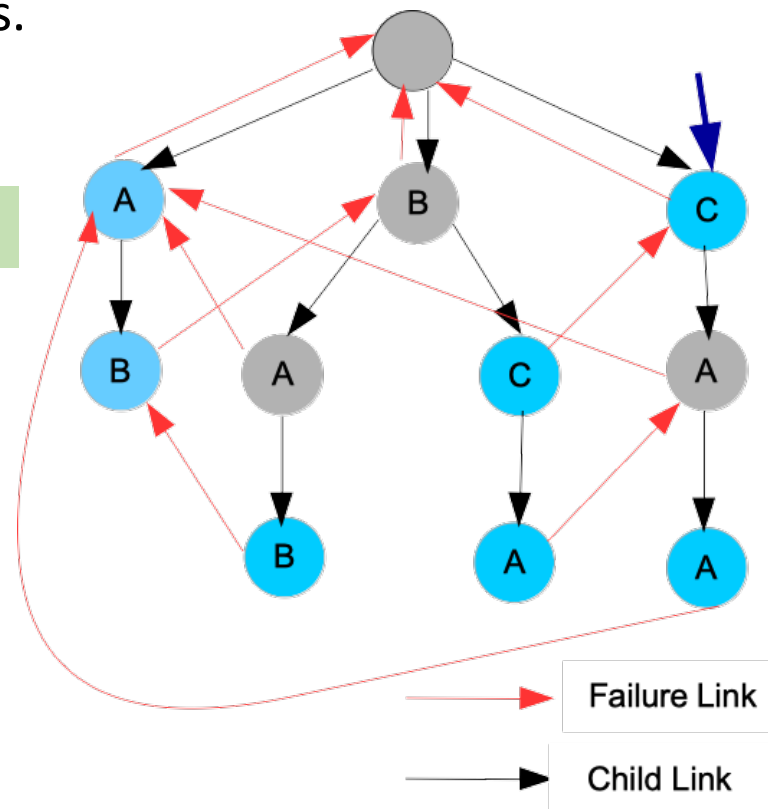
Match BC



Aho-Corasick

1. Build the Trie
2. Add failure links.

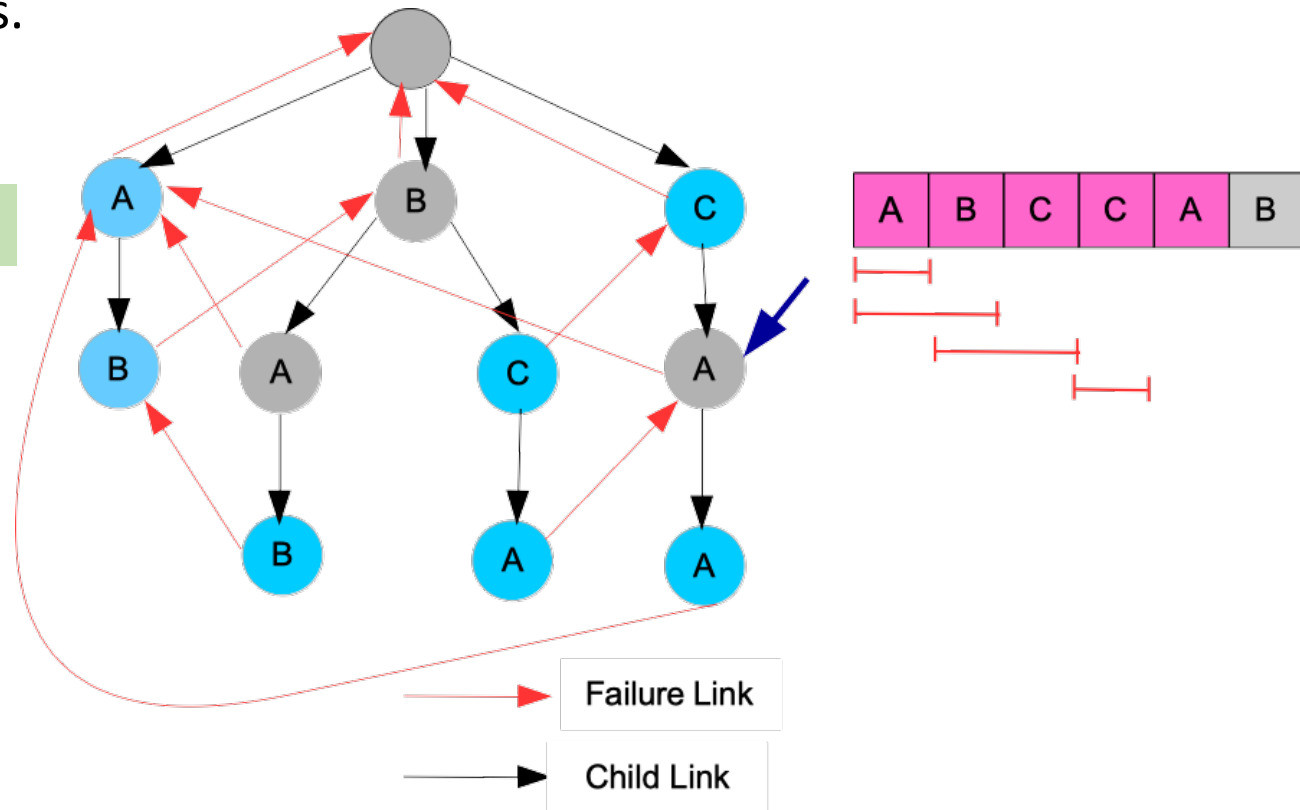
Match C



Aho-Corasick

1. Build the Trie
2. Add failure links.

Match nothing



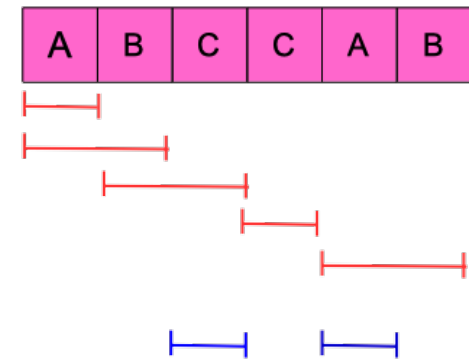
1. Build the Trie
2. Add failure links.

- ## Match AB

The diagram illustrates a hierarchical tree structure with nodes labeled A, B, and C. Nodes are represented by circles, with blue circles indicating active or current states and gray circles indicating failed or inactive states. Black arrows represent child links, and red arrows represent failure links. A blue arrow points to a specific node B, and a green bar highlights a portion of the tree.

Legend:

- Failure Link (Red Arrow)
- Child Link (Black Arrow)

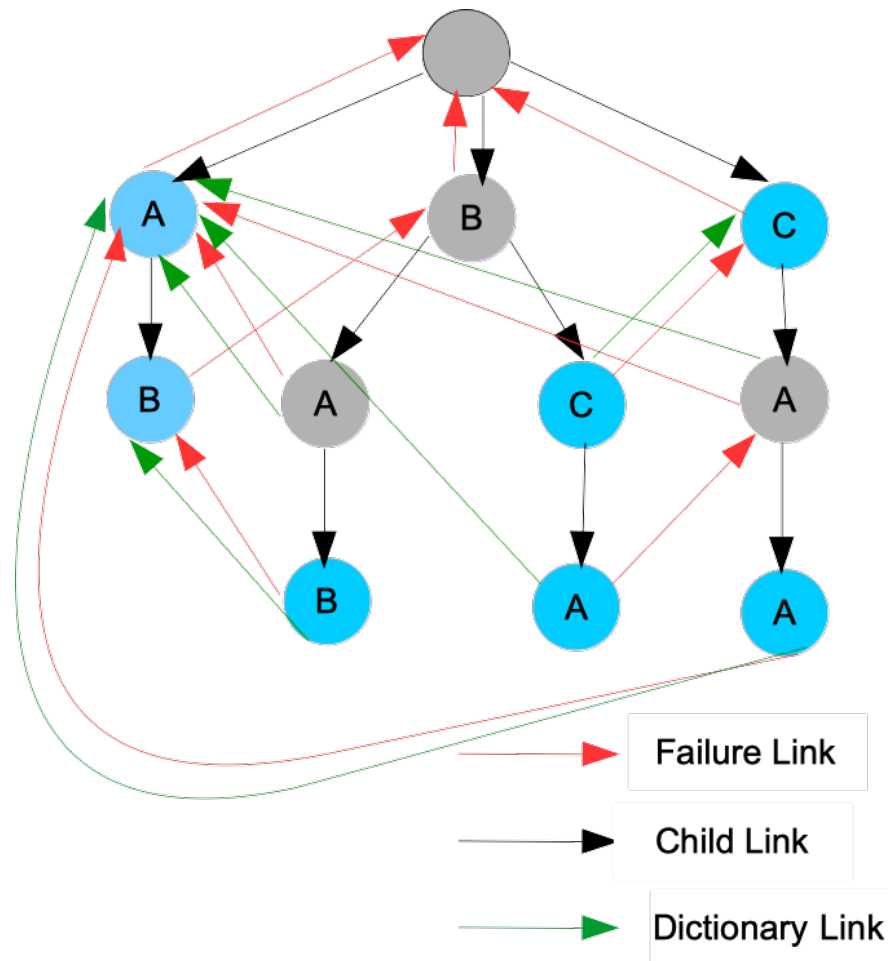


Aho-Corasick

1. Build the Trie
2. Add failure links.
3. Add dictionary links.

For each node, follow failure Links to until we hit root node or blue node.

$O(L(T) + M)$ where M is the number of matches



Experiments

- Randomly choose n patterns from roughly 10000 most common English words
- Randomly choose 10,000,000 words from all English words bank.

