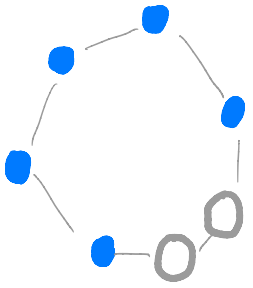
A New Approach In Finding Stable Coloring.

① How is A "Naive" Way in searching stable Coloring ?

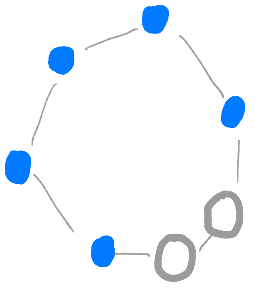
Recall: What is stable Coloring ?

Ex.



Recall: What is stable Coloring ?

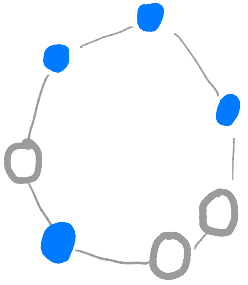
Ex.



Yes

Recall: What is stable Coloring ?

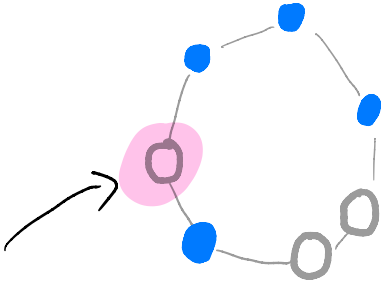
Ex.



No.

Recall: What is stable Coloring?

Ex.



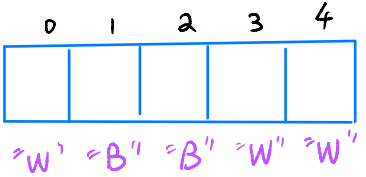
This node is solitary.

No.

A New Approach In Finding Stable Coloring.

① How is A "Naive" Way in searching stable Coloring ?

- Using Array (or List) to represent this structure graph.

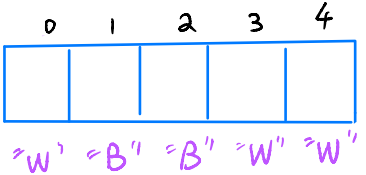


- "B" : coloring as blue
- "W" : coloring as white.

A New Approach In Finding Stable Coloring.

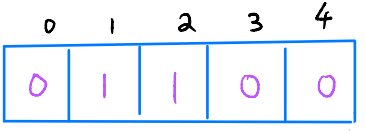
① How is A "Naive" Way in searching stable Coloring ?

- Using Array (or List) to represent this structure graph.



  - "B" : coloring as blue
  - "W" : coloring as white.

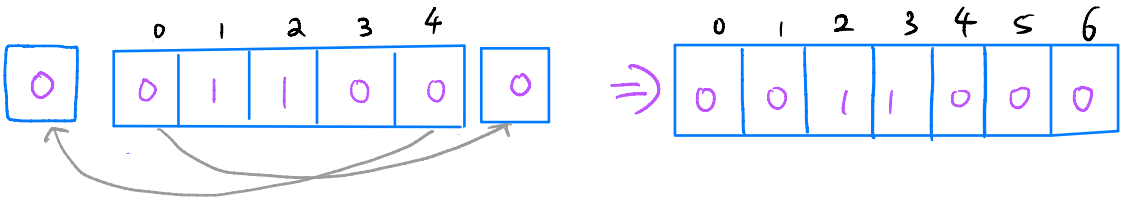
- We can further simplified with using integer "0" or "1".



  - "0" : coloring as white.
  - "1" : coloring as blue.

- Later, we should extend the Array (or List). Since, recall to the above example, node 0 and node 4 are neighbor to each other.

i.e.



- Then, define $f=[1,-2,1]$, and operate with rolling-convolution.

i.e. $[1,-2,1]$

$[0,0,1,1,0,0,0]$

$= 1\cdot 0 + -2\cdot 0 + 1\cdot 1$

$= 1$

$[1,-2,1]$

$[0,0,1,1,0,0,0]$, ...

$= 1\cdot 0 + (-2)(0) + 1\cdot 1$

$= 1$

- This is a iterative method. It does not need to store each calculated value while doing rolling-convolution.

- It would return "False" once there is a value == 2 or -2. which means there is a mode solitary.

$\Leftrightarrow$ This configuration of coloring is non-stable.

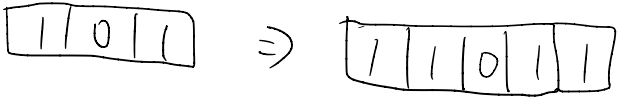
Analysis of This Algorithm.

- Give a pos. int N, generate all the binary strings of N bits.

  i.e. $n = 2 \Rightarrow \{[0,0], [0,1], [1,0], [1,1]\}$

  $\rightsquigarrow O(2^N)$

- For each binary strings, extends the front and back.

  i.e. 

  $\rightsquigarrow O(N)$

- Determine if there is node, within this cycle graph, being solitary

  $\rightsquigarrow O(N)$

  $\Rightarrow O(2^N)$

Can We Do It Better??

Can We Do It Better ??

Maybe!

Possible Idea :

- Suppose We had the "solution set" for the stable coloring of a cycle graph consisting with 4 nodes, and the set of 5 nodes as well.
- Note: We store the "solution set" as integer instead of N bits binary strings. i.e. "0011" = 3, "1111" = 7.

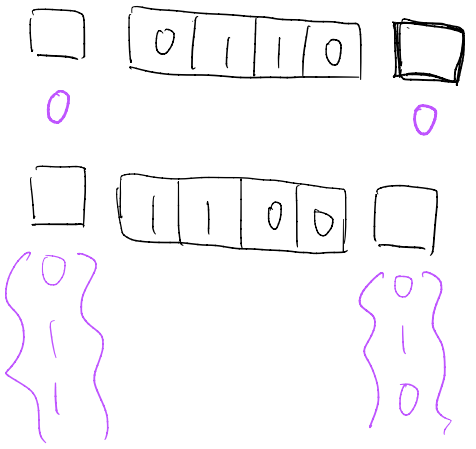
- In order to reduce # of operations, we should have the searching space as SMALL as possible.

- For example, we want to find the # configuration of stable coloring for cycle graph consisting with 6 nodes.
  → The searching space for this Problem is all 6-bit (possibles) binary string. ⇒ $2^6 = 64$.

- Can we limit our searching space ?

- **If** a mutation of a configuration of stable coloring on 4-node graph be a "reasonable" candidate on 6-node graph.

i.e.



- $\|S_4\|$, where S_4 is the "solution set" of 4-node graph.

- By the above mutation, we could have a searching space on $O(4\|S_4\|)$ for 6-node graph.

- By Thm 2.8, $\|S_4\| \approx \Phi^4 = 1.62^4 = 6$.

  $\Rightarrow 4 \cdot \|S_4\| \approx 4 \cdot 6 < 2^6 = 64$.