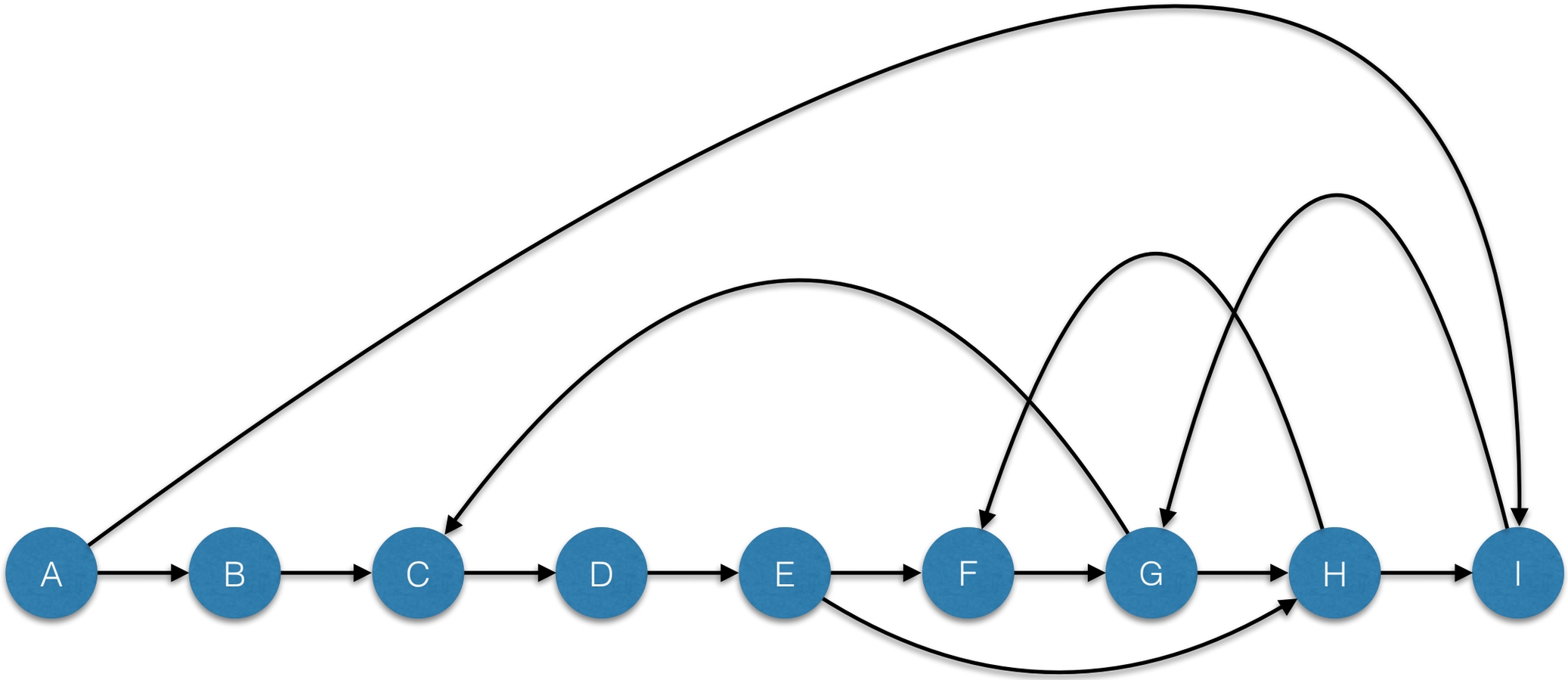
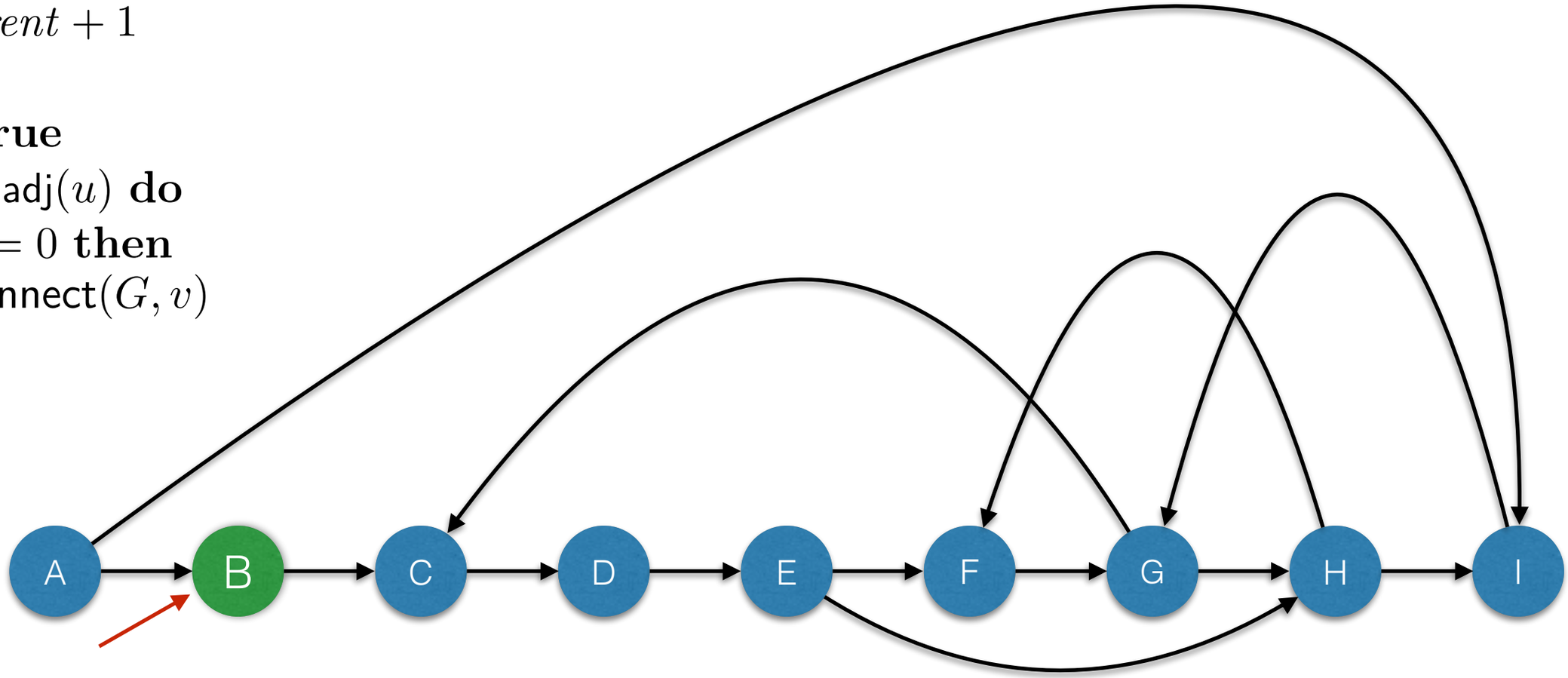


Index

LowLink

$index[u] \leftarrow lowlink[u] \leftarrow current$
$current \leftarrow current + 1$
$S.\mathsf{push}(u)$
$inStack[u] \leftarrow \mathbf{true}$
foreach $v \in G.\mathsf{adj}(u)$ **do**
  if $index[v] = 0$ **then**
    strongconnect(G, v)

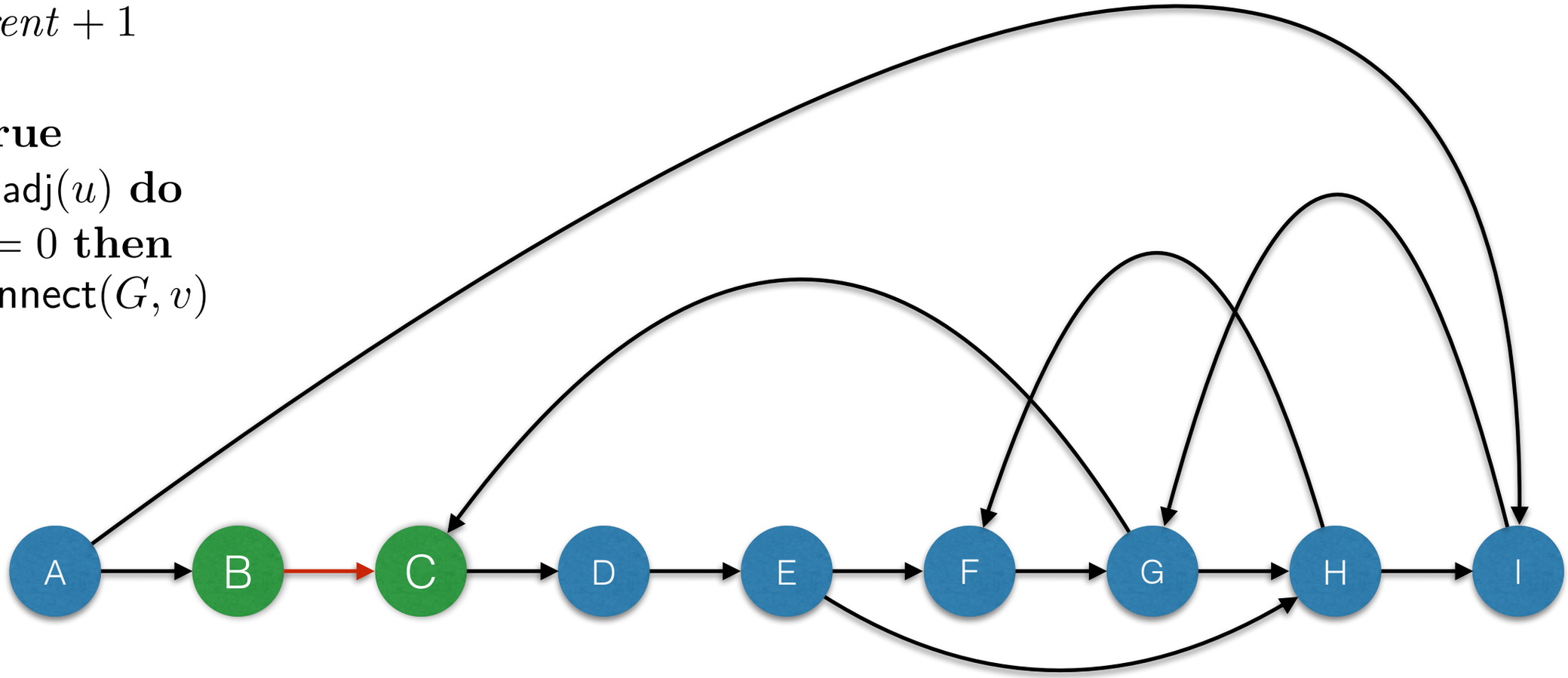


Index 1

LowLink 1

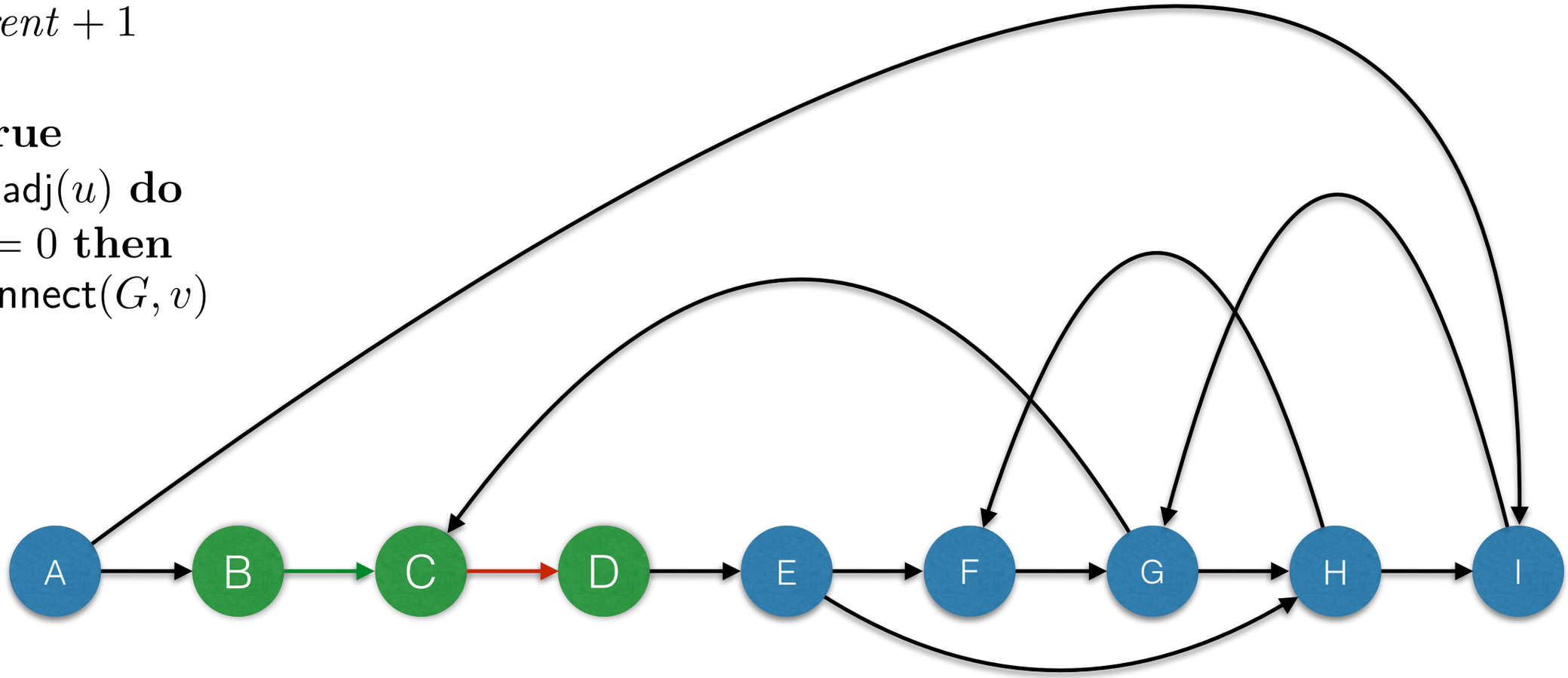
Stack B

$index[u] \leftarrow lowlink[u] \leftarrow current$
$current \leftarrow current + 1$
S.push(u)
$inStack[u] \leftarrow \textbf{true}$
foreach $v \in G$.adj(u) **do**
  if $index[v] = 0$ **then**
    strongconnect(G, v)



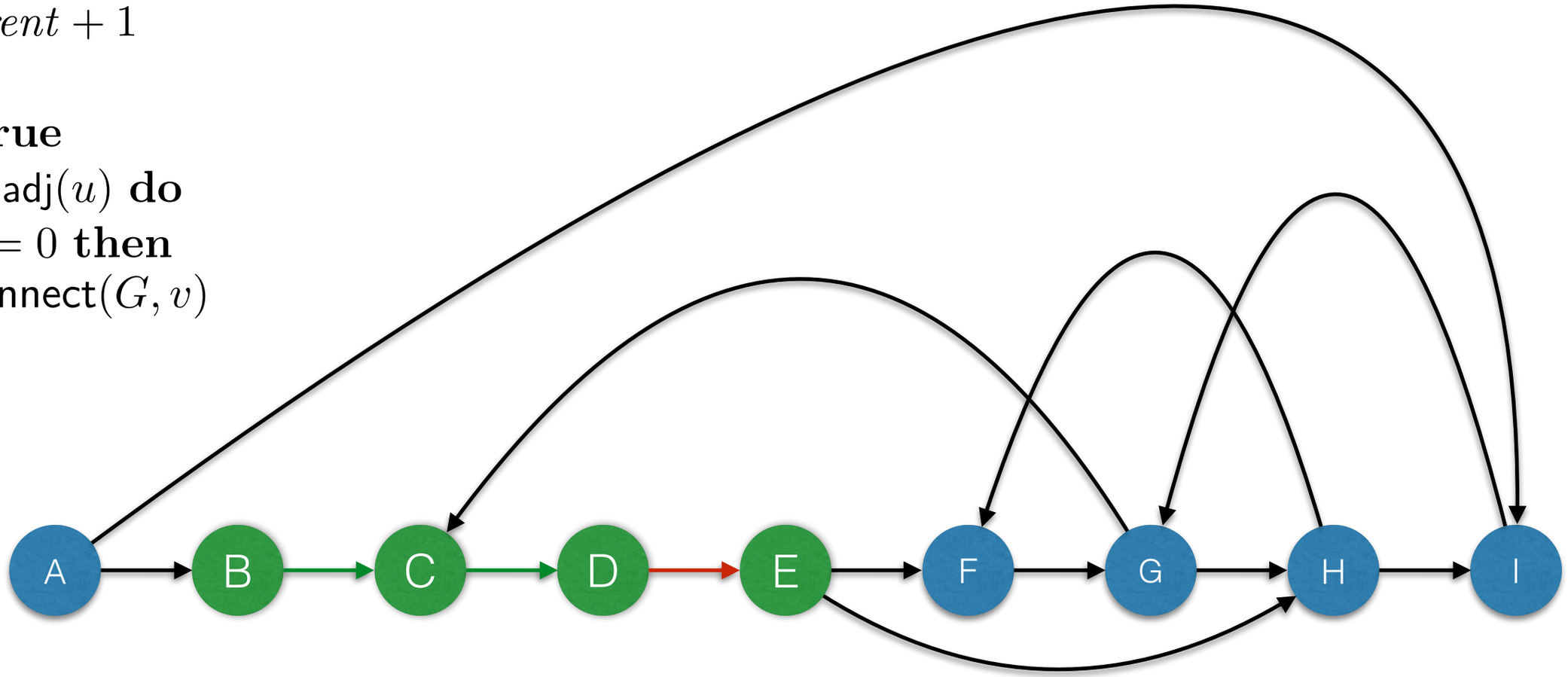
Index		1	2					
LowLink		1	2					
Stack	B	C						

$index[u] \leftarrow lowlink[u] \leftarrow current$
$current \leftarrow current + 1$
$S.\mathsf{push}(u)$
$inStack[u] \leftarrow \mathbf{true}$
foreach $v \in G.\mathsf{adj}(u)$ **do**
  if $index[v] = 0$ **then**
    strongconnect(G, v)



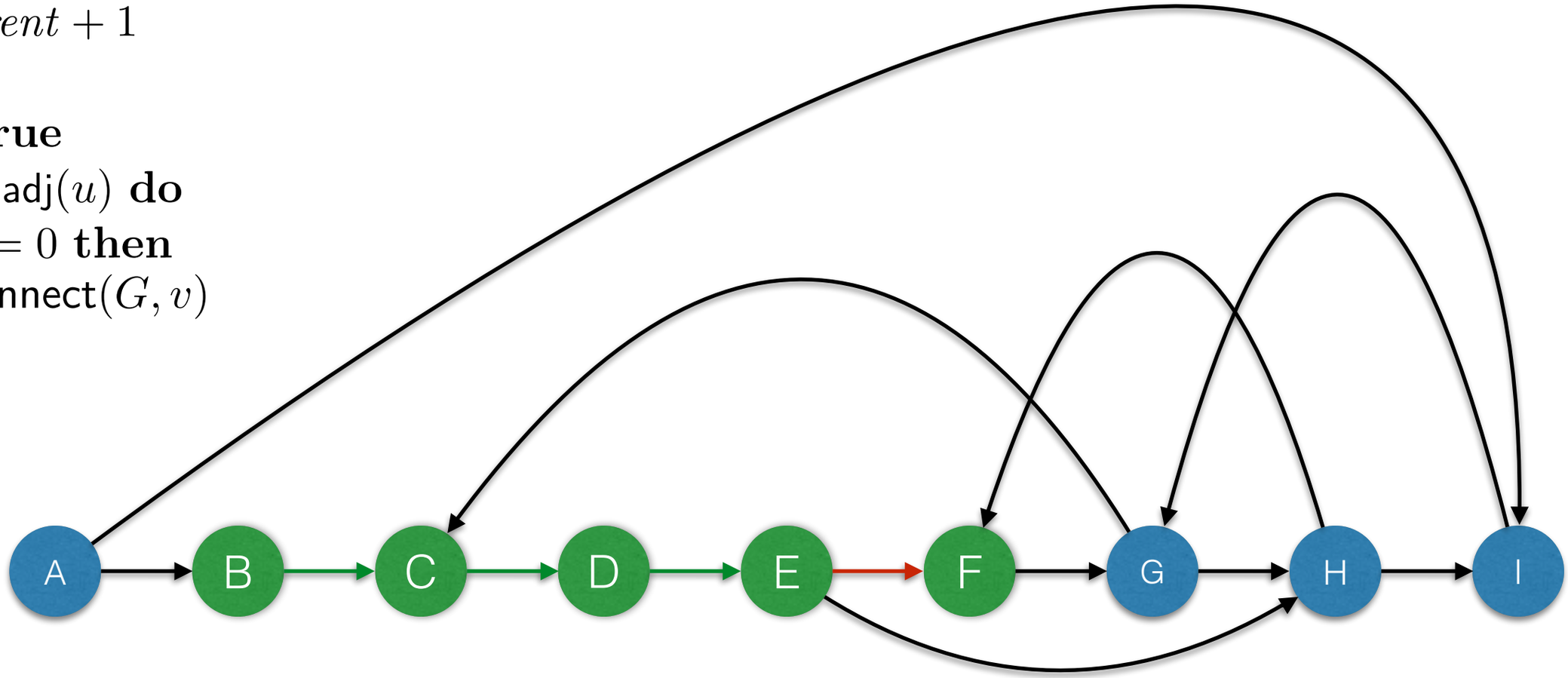
Index	1	2	3
LowLink	1	2	3
Stack	B	C	D

$index[u] \leftarrow lowlink[u] \leftarrow current$
$current \leftarrow current + 1$
$S.\mathsf{push}(u)$
$inStack[u] \leftarrow \mathbf{true}$
foreach $v \in G.\mathsf{adj}(u)$ **do**
  if $index[v] = 0$ **then**
    $\mathsf{strongconnect}(G, v)$



Index	1	2	3	4
LowLink	1	2	3	4
Stack	B	C	D	E

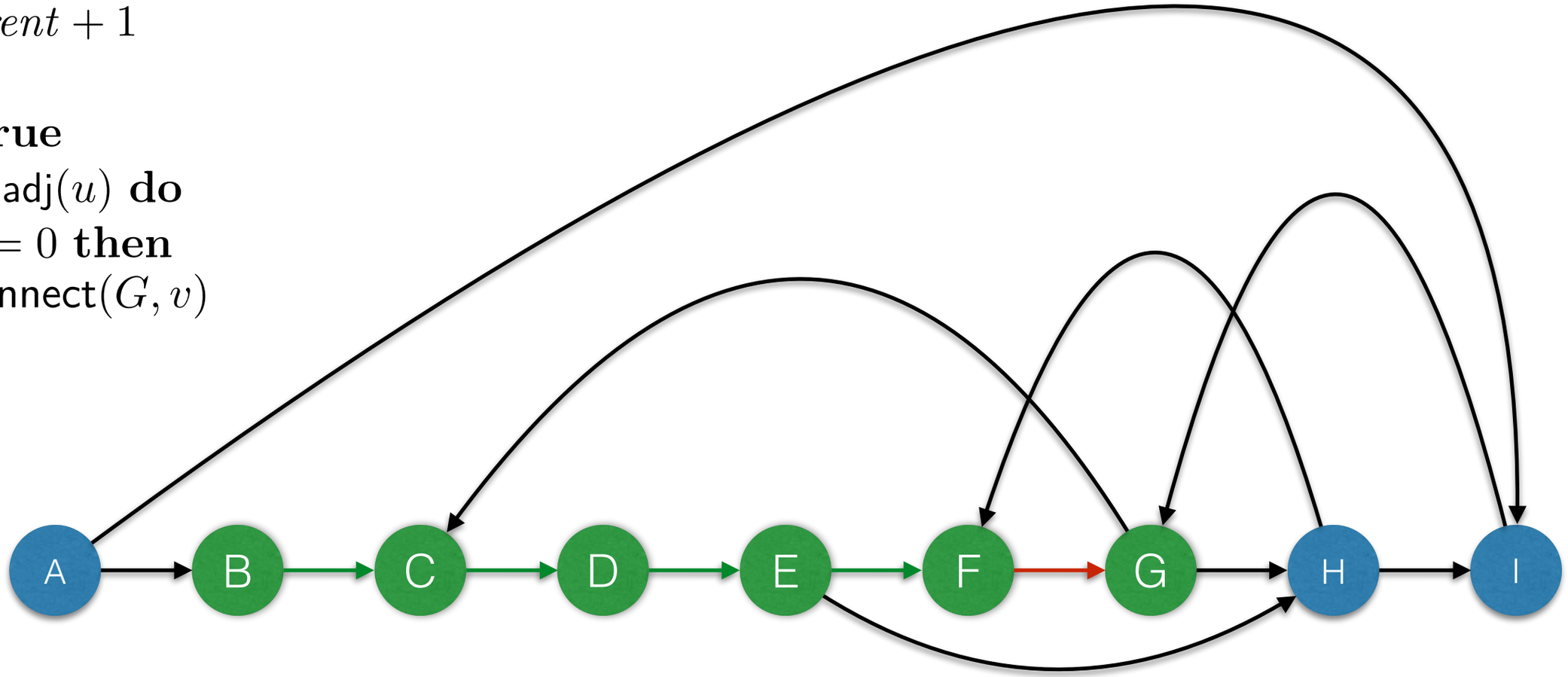
$index[u] \leftarrow lowlink[u] \leftarrow current$
$current \leftarrow current + 1$
$S.\mathsf{push}(u)$
$inStack[u] \leftarrow \mathbf{true}$
foreach $v \in G.\mathsf{adj}(u)$ **do**
  if $index[v] = 0$ **then**
    strongconnect(G, v)



Index	1	2	3	4	5
LowLink	1	2	3	4	5

Stack B C D E F

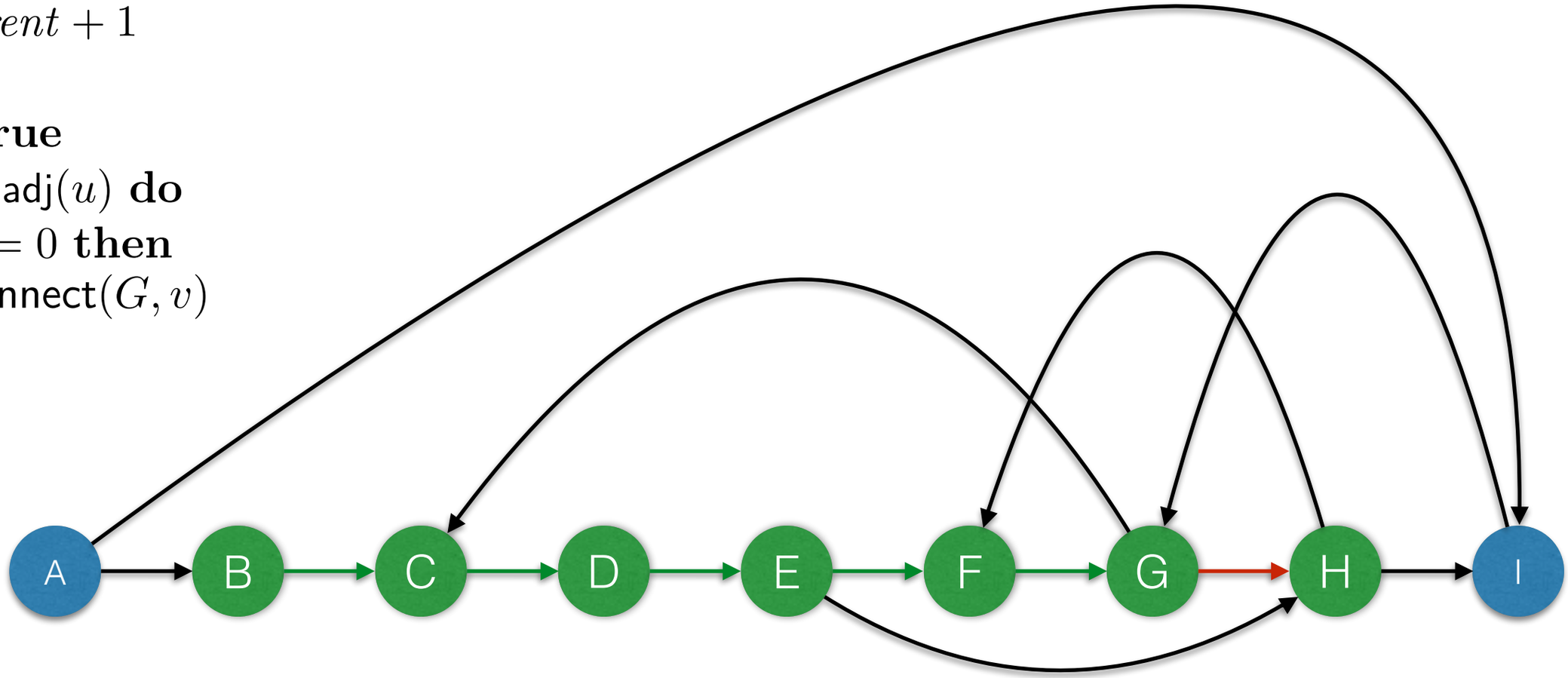
$index[u] \leftarrow lowlink[u] \leftarrow current$
$current \leftarrow current + 1$
$S.\mathsf{push}(u)$
$inStack[u] \leftarrow \mathbf{true}$
foreach $v \in G.\mathsf{adj}(u)$ **do**
  if $index[v] = 0$ **then**
    $\mathsf{strongconnect}(G, v)$



Index	1	2	3	4	5	6
LowLink	1	2	3	4	5	6

Stack B C D E F G

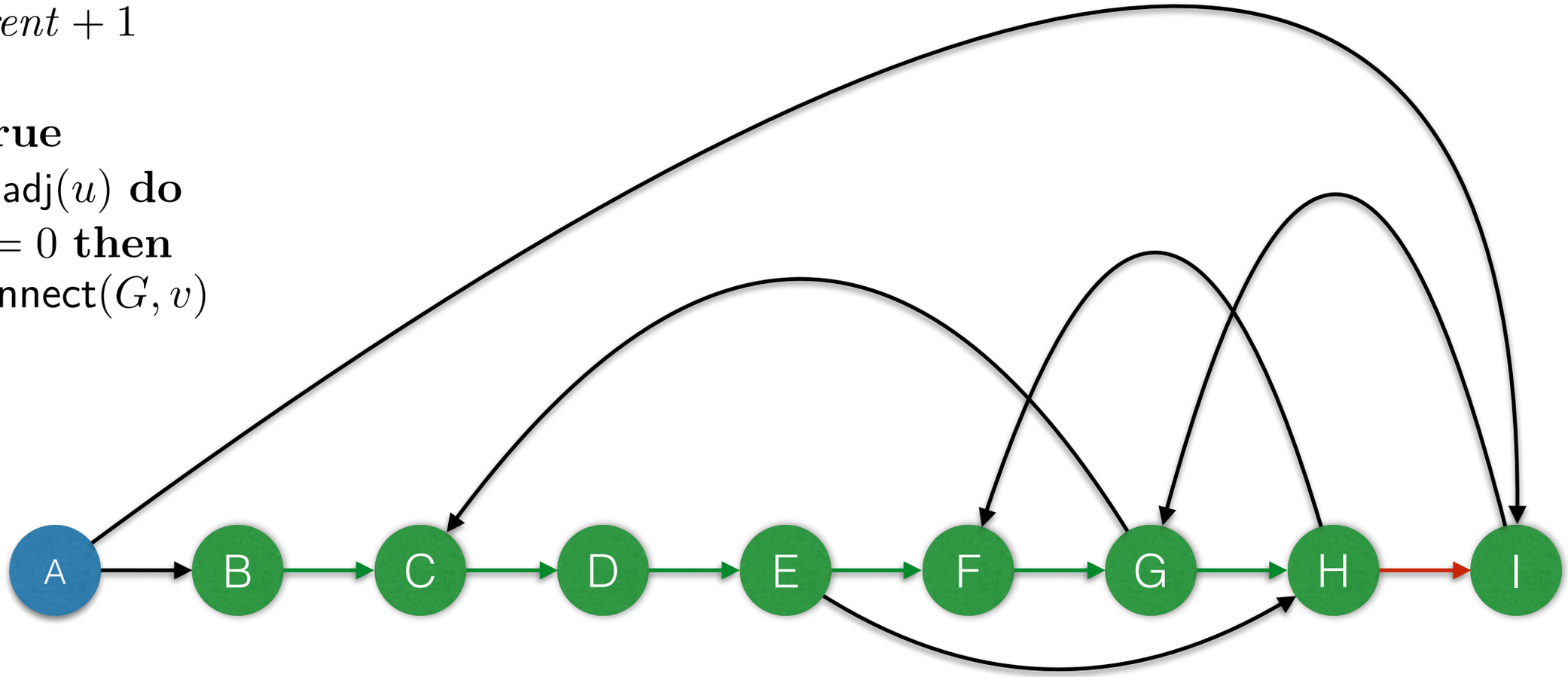
$index[u] \leftarrow lowlink[u] \leftarrow current$
$current \leftarrow current + 1$
$S.\mathsf{push}(u)$
$inStack[u] \leftarrow \mathbf{true}$
foreach $v \in G.\mathsf{adj}(u)$ **do**
  if $index[v] = 0$ **then**
    $\mathsf{strongconnect}(G, v)$



Index	1	2	3	4	5	6	7
LowLink	1	2	3	4	5	6	7

Stack B C D E F G H

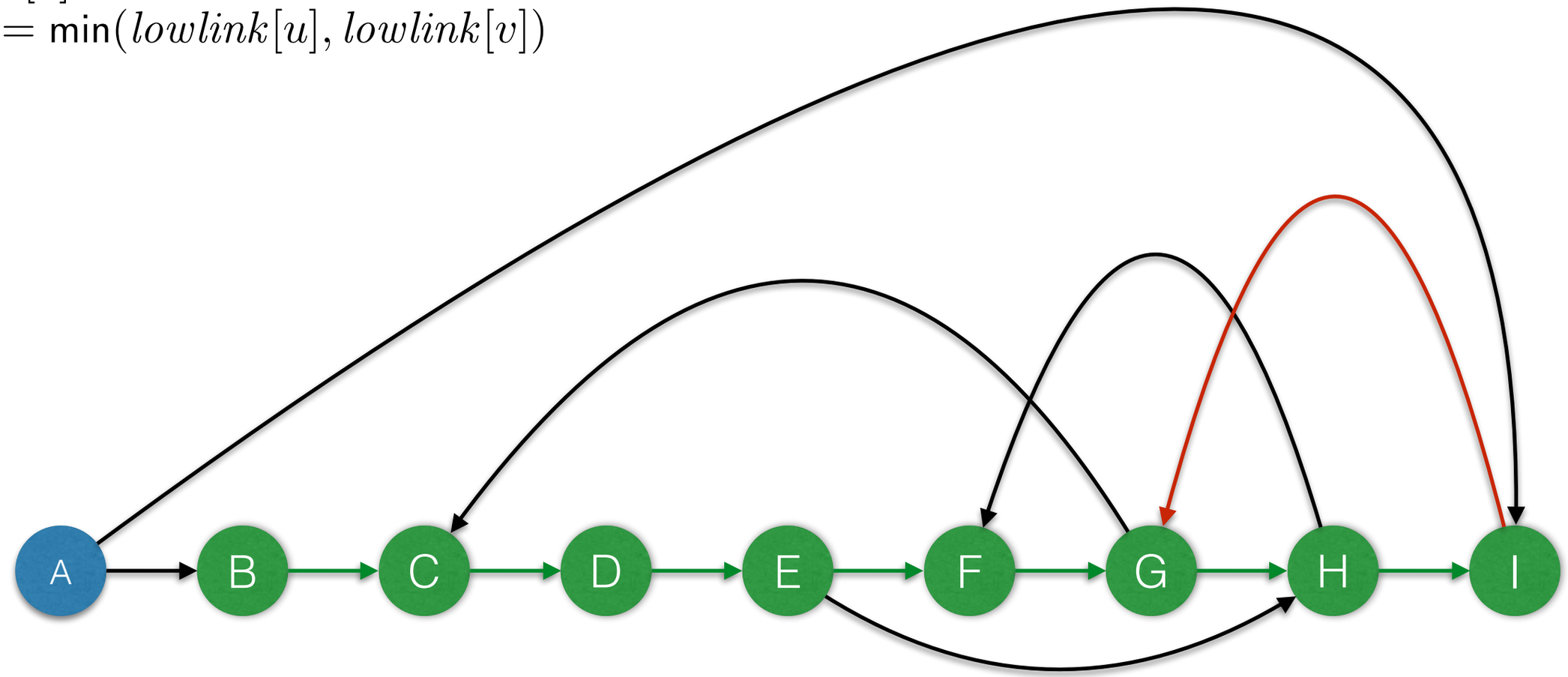
$index[u] \leftarrow lowlink[u] \leftarrow current$
$current \leftarrow current + 1$
$S.\mathsf{push}(u)$
$inStack[u] \leftarrow \mathbf{true}$
foreach $v \in G.\mathsf{adj}(u)$ **do**
  if $index[v] = 0$ **then**
    $\mathsf{strongconnect}(G, v)$



Index	1	2	3	4	5	6	7	8
LowLink	1	2	3	4	5	6	7	8

Stack	B	C	D	E	F	G	H	I

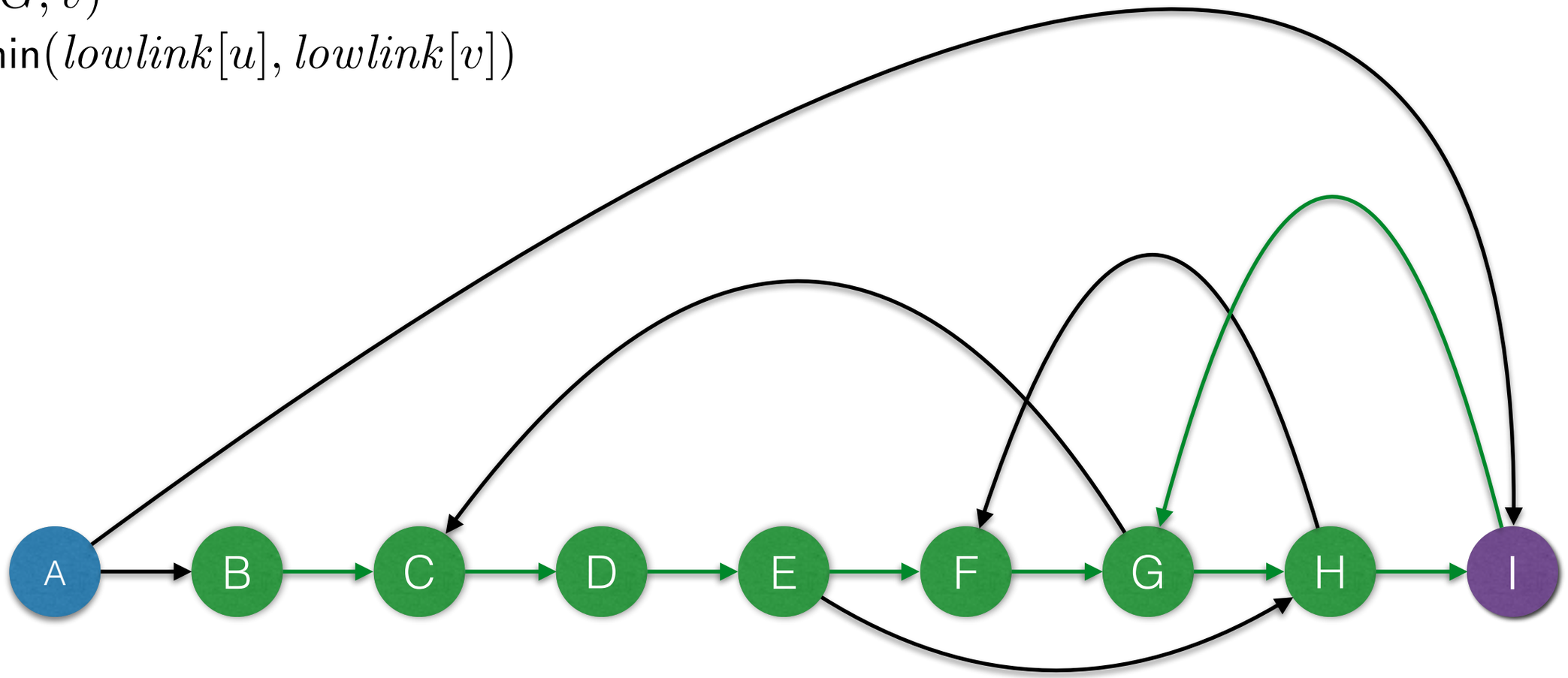
else if $inStack[v]$ **then**
  $lowlink[u] = \min(lowlink[u], lowlink[v])$



Index		1	2	3	4	5	6	7	8
LowLink		1	2	3	4	5	6	7	6
Stack	B	C	D	E	F	G	H	I	

strongconnect(G, v)

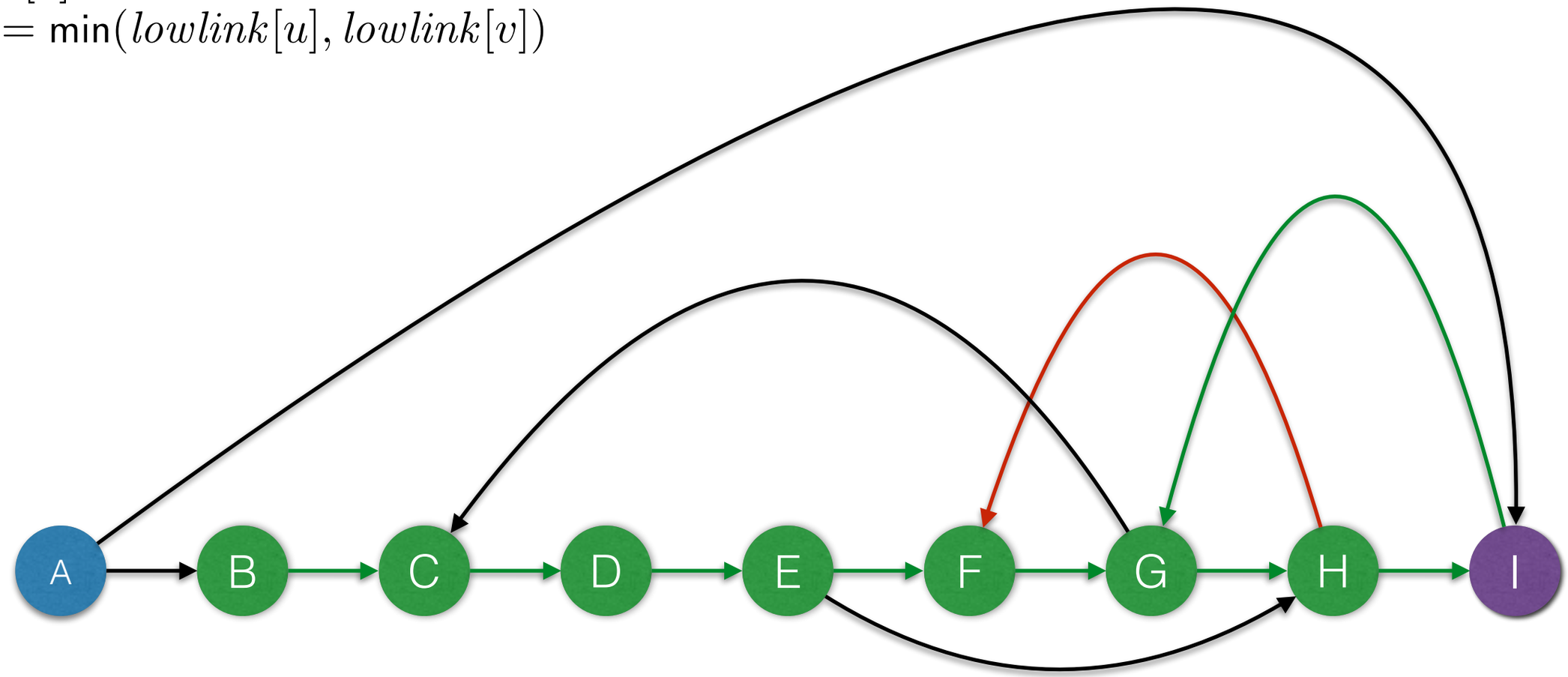
$lowlink[u] = \min(lowlink[u], lowlink[v])$



Index	1	2	3	4	5	6	7	8
LowLink	1	2	3	4	5	6	6	6

Stack B C D E F G H I

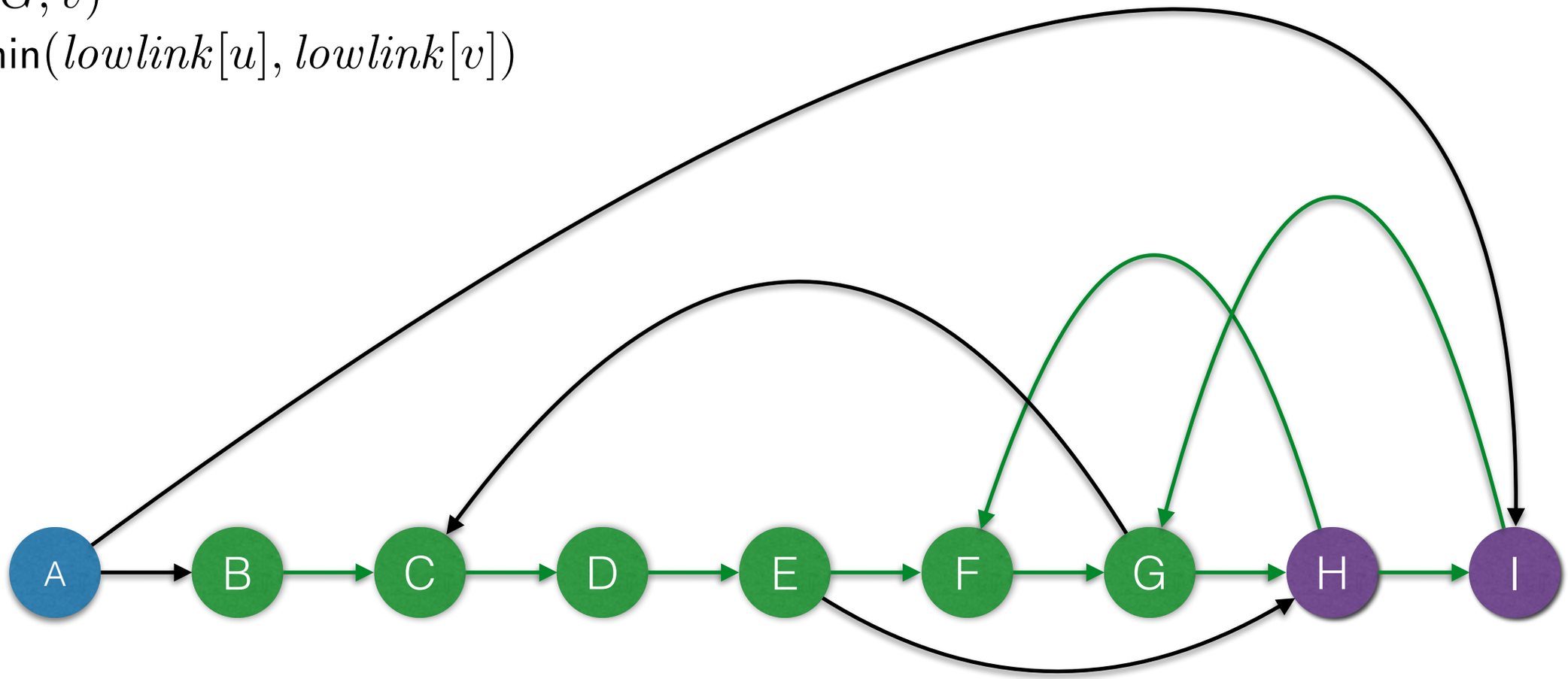
else if $inStack[v]$ **then**
  $lowlink[u] = \mathsf{min}(lowlink[u], lowlink[v])$



Index		1	2	3	4	5	6	7	8
LowLink		1	2	3	4	5	6	5	6
Stack	B	C	D	E	F	G	H	I	

strongconnect(G, v)

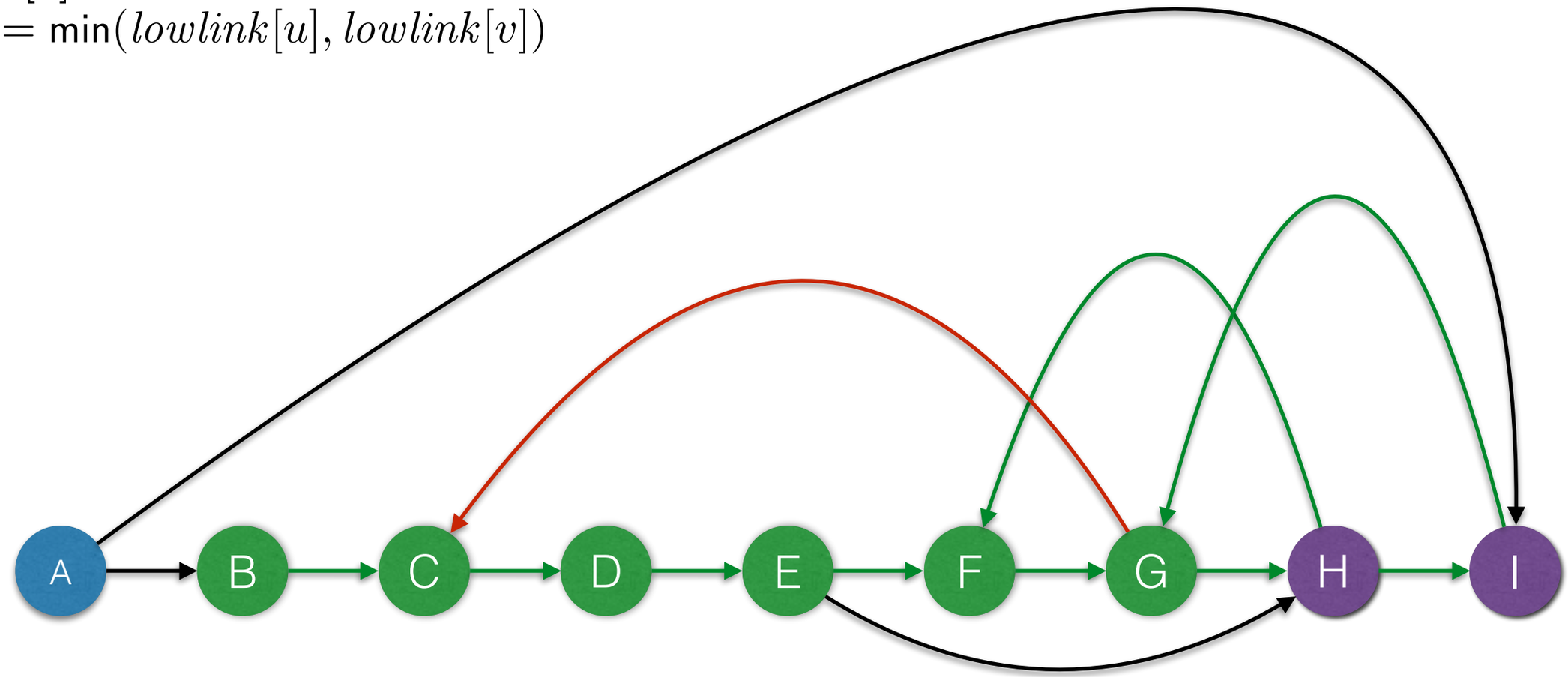
$lowlink[u] = \min(lowlink[u], lowlink[v])$



Index	1	2	3	4	5	6	7	8
LowLink	1	2	3	4	5	5	5	6

Stack	B	C	D	E	F	G	H	I

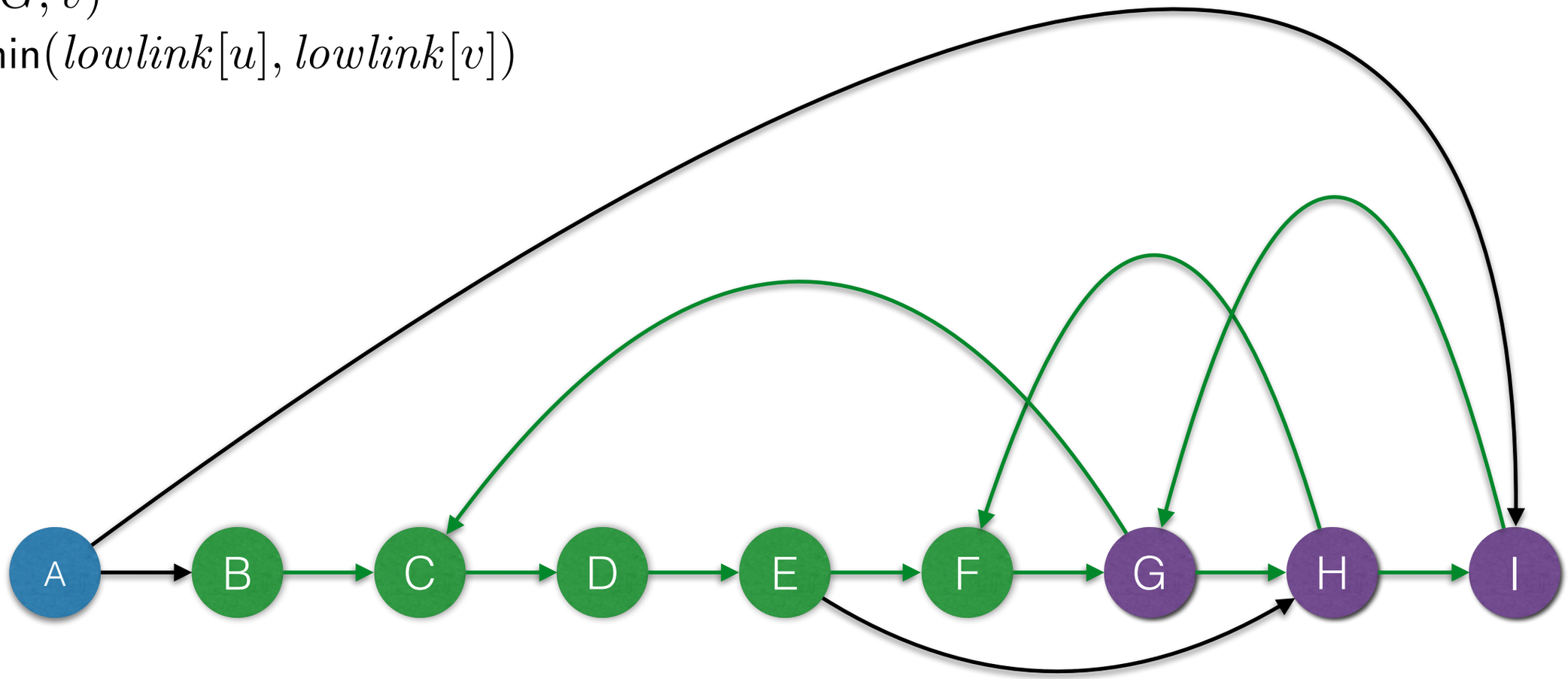
else if $inStack[v]$ **then**
  $lowlink[u] = \mathsf{min}(lowlink[u], lowlink[v])$



Index		1	2	3	4	5	6	7	8
LowLink		1	2	3	4	5	2	5	6
Stack	B	C	D	E	F	G	H	I	

strongconnect(G, v)

$lowlink[u] = \min(lowlink[u], lowlink[v])$

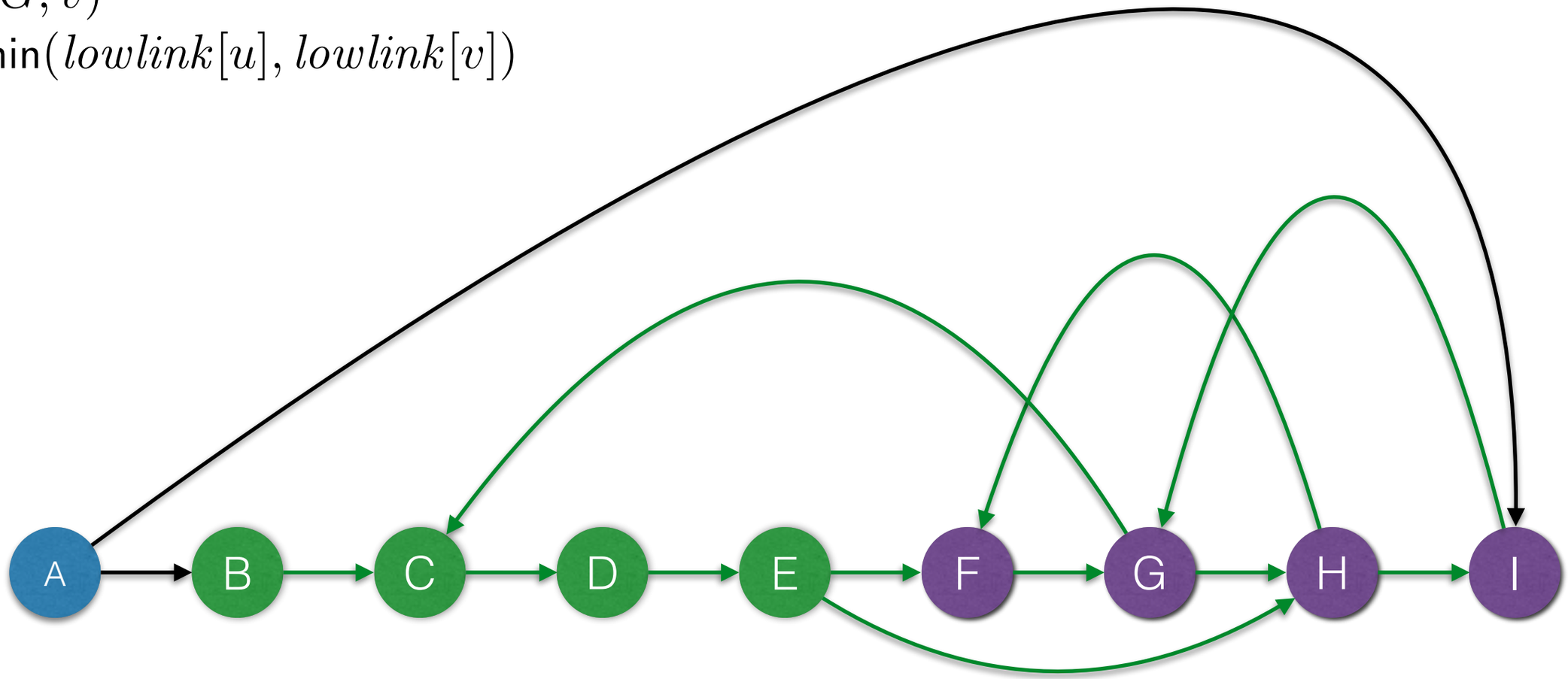


Index	1	2	3	4	5	6	7	8
LowLink	1	2	3	4	2	2	5	6

Stack	B	C	D	E	F	G	H	I

strongconnect(G, v)

$lowlink[u] = \min(lowlink[u], lowlink[v])$

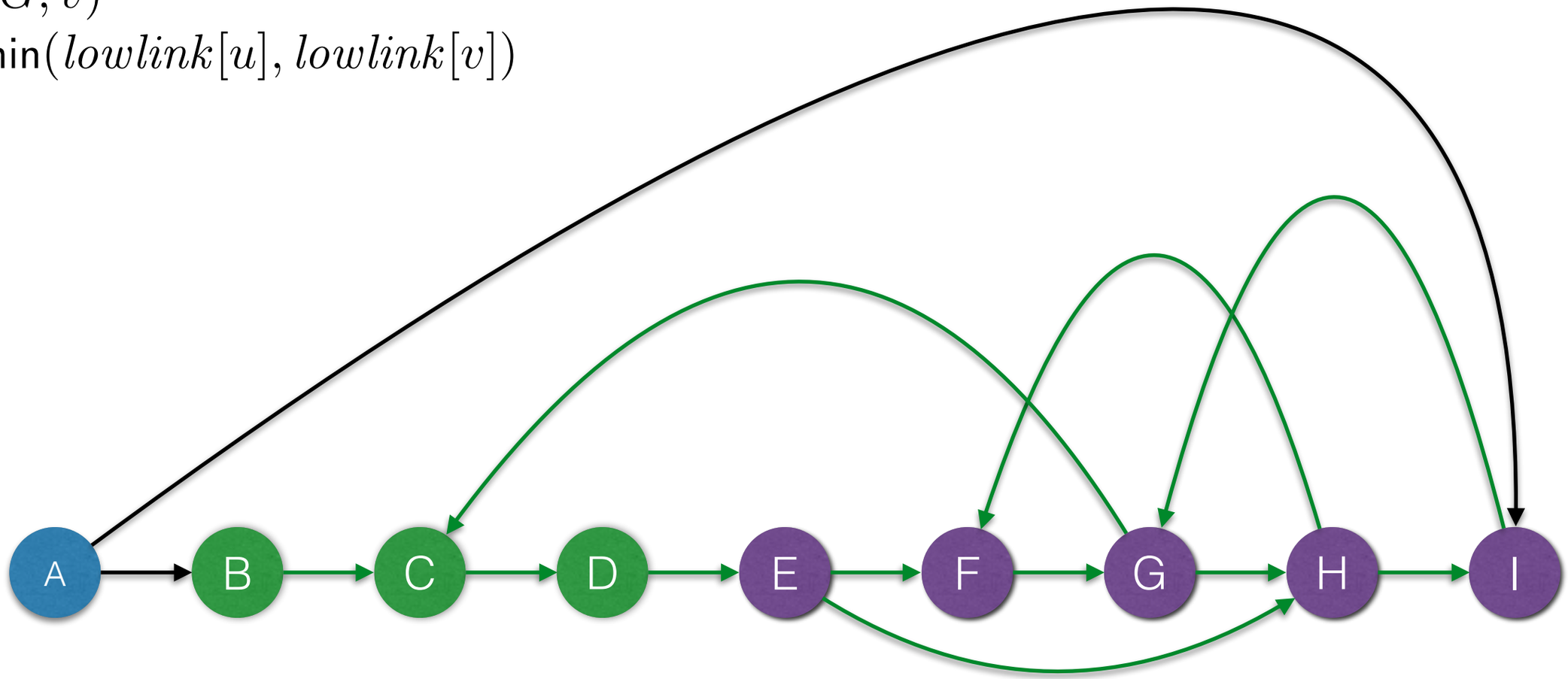


Index	1	2	3	4	5	6	7	8
LowLink	1	2	3	2	2	2	5	6

Stack B C D E F G H I

strongconnect(G, v)

$lowlink[u] = \min(lowlink[u], lowlink[v])$

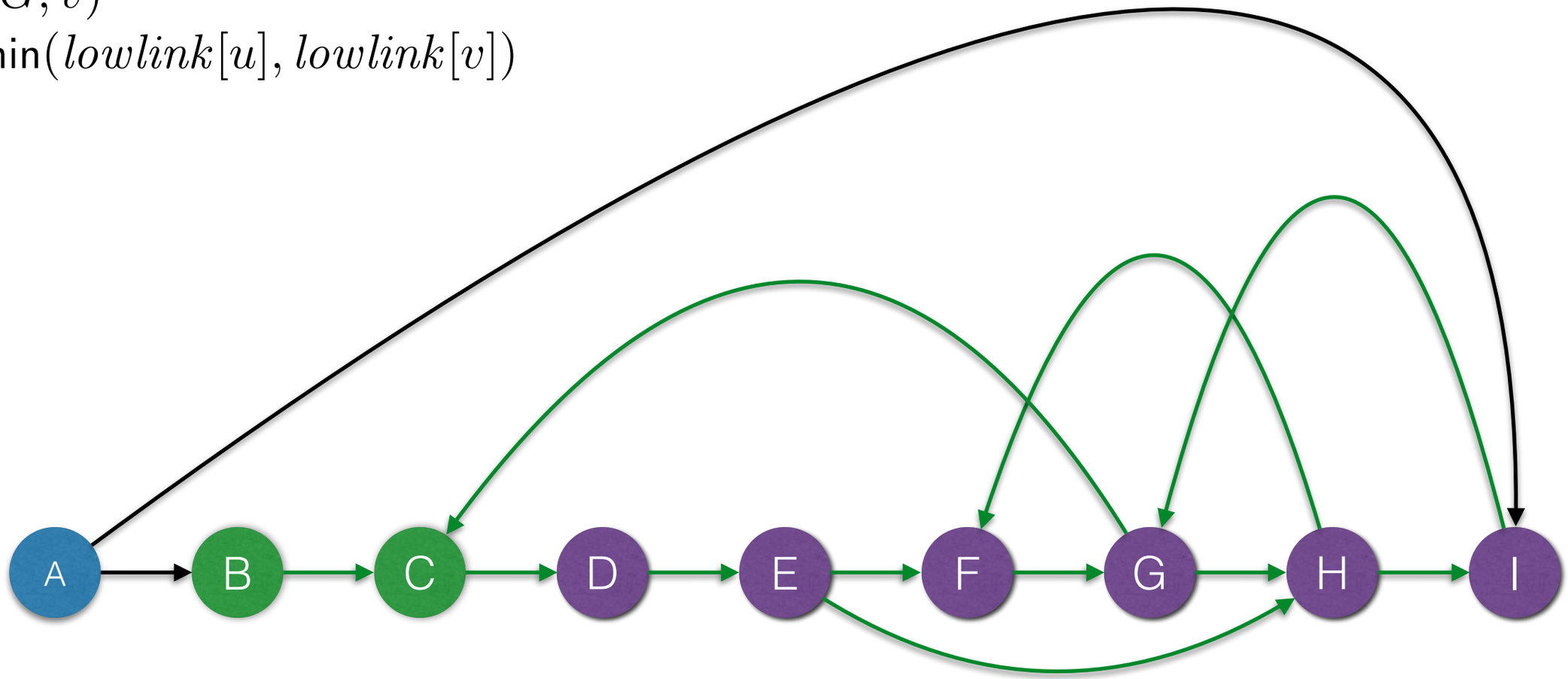


Index	1	2	3	4	5	6	7	8
LowLink	1	2	2	2	2	2	5	6

Stack	B	C	D	E	F	G	H	I

strongconnect(G, v)

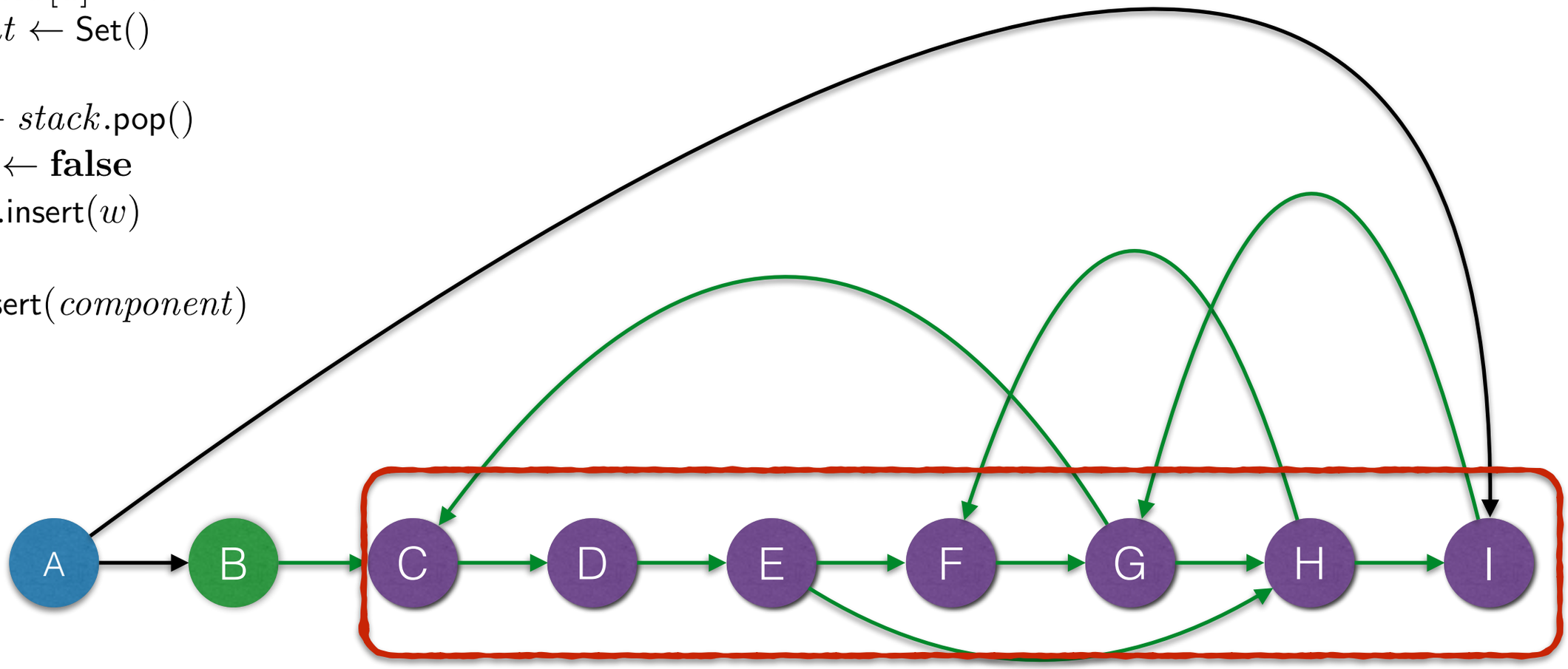
$lowlink[u] = \min(lowlink[u], lowlink[v])$



Index	1	2	3	4	5	6	7	8
LowLink	1	2	2	2	2	2	5	6

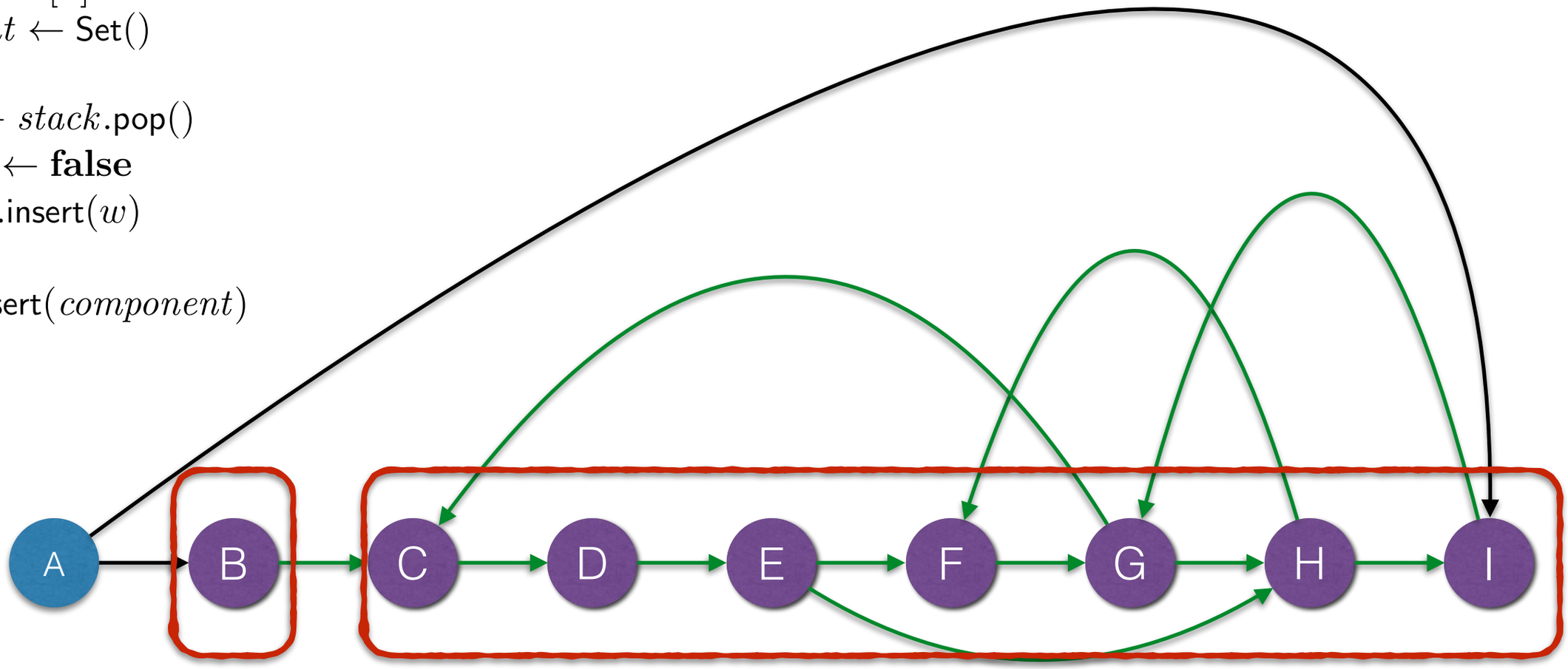
Stack	B	C	D	E	F	G	H	I

```
if lowlink[u] = index[u] then
    SET component ← Set()
    repeat
        NODE w ← stack.pop()
        inStack[w] ← false
        component.insert(w)
    until w = u
    components.insert(component)
```



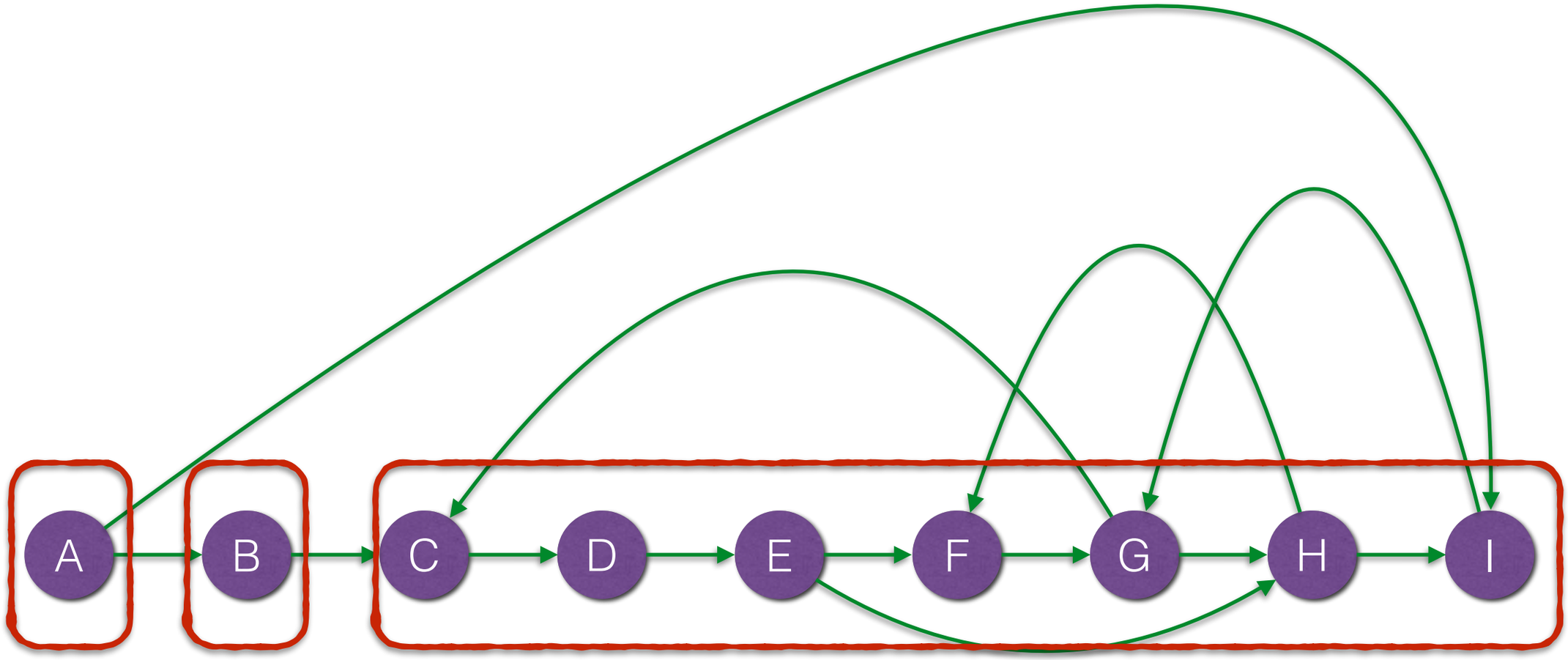
Index		1	2	3	4	5	6	7	8
LowLink		1	2	2	2	2	2	5	6
Stack	B								

```
if lowlink[u] = index[u] then
    SET component ← Set()
    repeat
        NODE w ← stack.pop()
        inStack[w] ← false
        component.insert(w)
    until w = u
    components.insert(component)
```



Index	1	2	3	4	5	6	7	8
LowLink	1	2	2	2	2	2	5	6

Stack



Index	9	1	2	3	4	5	6	7	8
LowLink	9	1	2	2	2	2	2	5	6

Stack