EXACT STRING MATCHING

EXACT STRING MATCHING

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]

EXACT STRING MATCHING

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]

PATTERN: AND
[1...n]
$n \leq m$

EXACT STRING MATCHING

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]

PATTERN: AND
[1...n]
n≤m

1) FIND P IN T or report that it's not there

2) FIND ALL MATCHES

EXACT STRING MATCHING

FIND ALL MATCHES

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]

PATTERN: AND
[1...n]
n≤m

Brute force: scan T, looking for P(1)

EXACT STRING MATCHING

FIND ALL MATCHES

Brute force: scan T, looking for P(1)

EXACT STRING MATCHING

FIND ALL MATCHES

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]

PATTERN: AND
[1...n]
n≤m

Brute force: scan T, looking for P(1)

EXACT STRING MATCHING

FIND ALL MATCHES

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]

PATTERN: AND
[1...n]
n≤m

Brute force: scan T, looking for P(1)

EXACT STRING MATCHING

FIND ALL MATCHES

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]

PATTERN: AND
[1...n]
n≤m

Brute force: scan T, looking for P(1)

EXACT STRING MATCHING

FIND ALL MATCHES

Brute force: scan T, looking for P(1)

↳ if T(i) matches P(1)

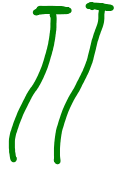
scan from T(i+1) looking for P(2)

etc

EXACT STRING MATCHING

FIND ALL MATCHES

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]



PATTERN: AND
[1...n]
n≤m

Brute force: scan T, looking for P(1)
↳ if T(i) matches P(1)
scan from T(i+1) looking for P(2)
etc

EXACT STRING MATCHING

FIND ALL MATCHES

Brute force: scan T, looking for P(1)

↳ if T(i) matches P(1)

scan from T(i+1) looking for P(2)

etc

How bad?

EXACT STRING MATCHING

FIND ALL MATCHES

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]

PATTERN: AND
[1...n]
n≤m

Brute force: scan T, looking for P(1)
↳ if T(i) matches P(1)
scan from T(i+1) looking for P(2)
etc

$O(n \cdot (m-n)) = O(m^2)$

T: XXXXXXXXXXXXXXXX

P: XXXXXXXO

EXACT STRING MATCHING

FIND ALL MATCHES

TEXT: ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY
[1...m]

PATTERN: AND
[1...n]
n≤m

Brute force: scan T, looking for P(1)
↳ if T(i) matches P(1)
scan from T(i+1) looking for P(2)
etc

$O(n \cdot (m-n)) = O(m^2)$

We will get $\Theta(n+m)$

T: XXXXXXXXXXXXXXXXX

P: XXXXXXXO

We will get $\Theta(n+m)$

Phase 1: process P in O(n) time

																					n
i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B

extention of example from Gusfield book

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i > 1)$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/																					

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i > 1)$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1																				

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i > 1)$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0																			

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i > 1)$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3																		

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i > 1)$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1																	

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i > 1)$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0													

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i > 1)$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7												

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i > 1)$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2																			

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6																		

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6																	

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8														

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16					

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ← (arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ← (arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2																				

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ← (arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2																			

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ←(arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4																		

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ← (arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4															

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ← (arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8													

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ← (arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10					

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ← (arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18				

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ← (arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

We will get $\Theta(n+m)$ Phase 1: process P in $O(n)$ time

Z_i = | longest string starting at P(i) & matching prefix of P | $(i>1)$

Z-box(i) = interval from i to $i+Z_i-1$ (for $Z_i>0$)

r_i = for all Z-boxes beginning at or before i, rightmost endpoint

l_i = left endpoint of any Z-box that ends at r_i ← (arbitrary)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Phase 1: Compute all Z_i of P in O(n) time

i:		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:		A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Phase 1: Compute all Z_i of P in $O(n)$ time

↳ get each Z_k in $O(1)$ using only r_{k-1} & l_{k-1}

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Phase 1: Compute all Z_i of P in $O(n)$ time

↳ get each Z_k in $O(1)$ using only r_{k-1} & l_{k-1}

... except, start by brute force for $i=2$

↳ if no match, $Z_2 = r = l = 0$

else $r = Z_2 + 1$ & $l = 2$ } A A A A A X, Z_2

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$.

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1																				
r	/	2																				
l	/	2																				

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$

No Z-box jumps over k.

Case 2: $k \leq r$

Z-box contains k

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1																				
r	/	2																				
l	/	2																				

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$

No Z-box jumps over k.

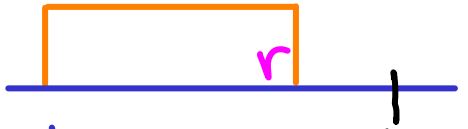
See if a new Z-box starts at k

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1																				
r	/	2																				
l	/	2																				

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$



No Z-box jumps over k.

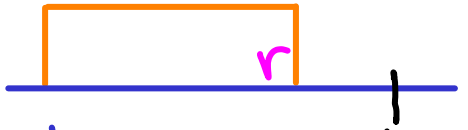
See if a new Z-box starts at k: Match P[k...] to prefix (brute force)

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	?																			
r	/	2	?																			
l	/	2	?																			

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$



No Z-box jumps over k.

See if a new Z-box starts at k: Match P[k...] to prefix (brute force)

If match = ∅ go to k+1.

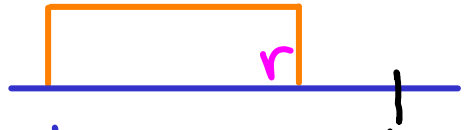
i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0																			
r	/	2	2																			
l	/	2	2																			

} inherited

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$



No Z-box jumps over k.

See if a new Z-box starts at k: Match P[k...] to prefix (brute force)

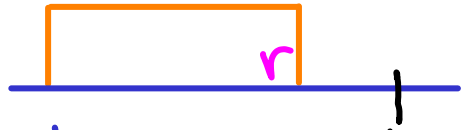
If match = ∅ go to k+1.

i:	1	2	3	4	5	6	k=7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	?															
r	/	2	2	6	6	6	?															
l	/	2	2	4	4	4	?															

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$



No Z-box jumps over k.

See if a new Z-box starts at k: Match P[k...] to prefix (brute force)

If match = ∅ go to k+1.

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0															
r	/	2	2	6	6	6	6															
l	/	2	2	4	4	4	4															

} inherited

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$

No Z-box jumps over k.

See if a new Z-box starts at k: Match P[k...] to prefix (brute force)

If match = ∅ go to k+1. Else ... ?

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	?												
r	/	2	2	6	6	6	6	8	8	?												
l	/	2	2	4	4	4	4	8	8	?												

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$

No Z-box jumps over k.

See if a new Z-box starts at k: Match P[k...] to prefix (brute force)

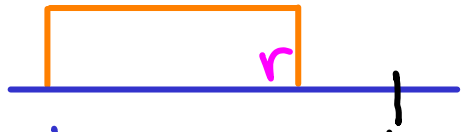
If match = ∅ go to k+1. Else $l = k$, $r = l + Z_k$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7												
r	/	2	2	6	6	6	6	8	8	16												
l	/	2	2	4	4	4	4	8	8	10												

Phase 1: Compute all Z_i of P in $O(n)$ time

↳ get each Z_k in $O(1)$ using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. **Case 1:** $k > r$



No Z-box jumps over k.

See if a new Z-box starts at k: Match $P[k\ldots]$ to prefix (brute force)

If match $= \emptyset$ go to $k+1$. Else $l = k$, $r = l + Z_k$

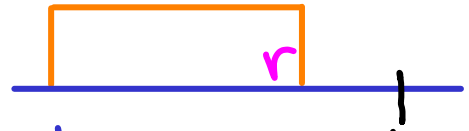
i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7												
r	/	2	2	6	6	6	6	8	8	16												
l	/	2	2	4	4	4	4	8	8	10												

Case 1 finds Z_k in time $\Theta(Z_k)$.

Phase 1: Compute all Z_i of P in $O(n)$ time

↳ get each Z_k in $O(1)$ using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$



No Z-box jumps over k.

See if a new Z-box starts at k: Match $P[k\ldots]$ to prefix (brute force)

If match $= \emptyset$ go to $k+1$. Else $l = k$, $r = l + Z_k$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7												
r	/	2	2	6	6	6	6	8	8	16												
l	/	2	2	4	4	4	4	8	8	10												

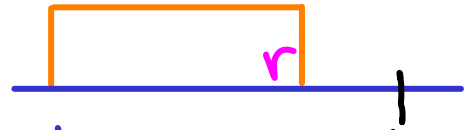
Case 1 finds Z_k in time $\Theta(Z_k)$.

It takes $> Z_k$ iterations to return to Case 1

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 1: $k > r$



No Z-box jumps over k.

See if a new Z-box starts at k: Match P[k...] to prefix (brute force)

If match = ∅ go to k+1. Else $l = k$, $r = l + Z_k$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7												
r	/	2	2	6	6	6	6	8	8	16												
l	/	2	2	4	4	4	4	8	8	10												

Case 1 finds Z_k in time $\Theta(Z_k)$.

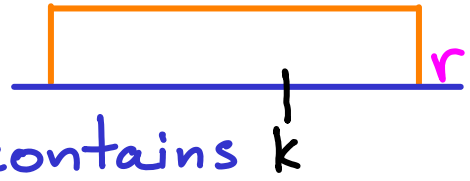
It takes $> Z_k$ iterations to return to Case 1

↳ time for all Case 1 in Phase 1: O(n)

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. **Case 2:** $k \leq r$

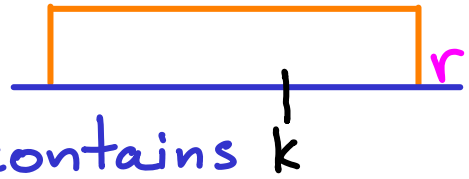


i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$



i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0										
r	/	2	2	6	6	6	6	8	8	16	16	16										
l	/	2	2	4	4	4	4	8	8	10	10	10										

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0										
r	/	2	2	6	6	6	6	8	8	16	16	16										
l	/	2	2	4	4	4	4	8	8	10	10	10										

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

β = suffix of Z-box
↳ starting at k

β = AABC

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0										
r	/	2	2	6	6	6	6	8	8	16	16	16										
l	/	2	2	4	4	4	4	8	8	10	10	10										

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step k-1 ≥ 2. Case 2: k ≤ r

Z-box contains k

↳ matches prefix

β = suffix of Z-box starting at k

must = β

k

β = AABC

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0										
r	/	2	2	6	6	6	6	8	8	16	16	16										
l	/	2	2	4	4	4	4	8	8	10	10	10										

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

k' must = β

k β = AABC

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0										
r	/	2	2	6	6	6	6	8	8	16	16	16										
l	/	2	2	4	4	4	4	8	8	10	10	10										

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step k-1 ≥ 2.

Case 2a: $Z(k') < |\beta|$ $|\beta| = r-k+1$

Case 2: $k \le r$

Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳starting at k

$k' = k-l+1$

k' must = β

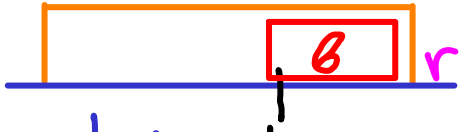
k β = AABC

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0										
r	/	2	2	6	6	6	6	8	8	16	16	16										
l	/	2	2	4	4	4	4	8	8	10	10	10										

Phase 1: Compute all Z_i of P in $O(n)$ time

↳ get each Z_k in $O(1)$ using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$



Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

Case 2a: $Z(k') < |\beta|$ $\quad |\beta| = r - k + 1$

↳ $Z_k = Z_{k'}$

must = β $\qquad$ β = AABC

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3									
r	/	2	2	6	6	6	6	8	8	16	16	16										
l	/	2	2	4	4	4	4	8	8	10	10	10										

Phase 1: Compute all Z_i of P in O(n) time

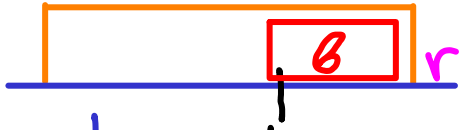
↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$.

Case 2a: $Z(k') < |\beta|$ $\quad |\beta| = r-k+1$

↳ $Z_k = Z_{k'}$ & r, l don't change

Case 2: $k \leq r$



Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k-l+1$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3									
r	/	2	2	6	6	6	6	8	8	16	16	16	16									
l	/	2	2	4	4	4	4	8	8	10	10	10	10									

k' must = β k β = AABC

Phase 1: Compute all Z_i of P in $O(n)$ time

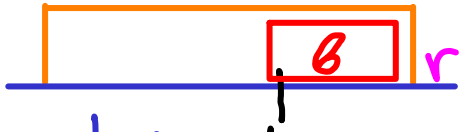
↳ get each Z_k in $O(1)$ using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$.

Case 2a: $Z(k') < |\beta|$ $\quad |\beta| = r-k+1$

↳ $Z_k = Z_{k'}$ & r, l don't change

Case 2: $k \leq r$



Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k-l+1$

$\beta = AABC$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3									
r	/	2	2	6	6	6	6	8	8	16	16	16	16									
l	/	2	2	4	4	4	4	8	8	10	10	10	10									

$O(1)$ time, go to $k+1$

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step k-1 ≥ 2. Case 2: k ≤ r

Z-box contains k

↳ matches prefix

β = suffix of Z-box

↳ starting at k

$k' = k - l + 1$

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r - k + 1$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19			

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

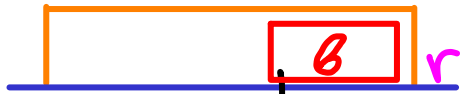
Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r - k + 1$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19			

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step k-1 ≥ 2. Case 2: k ≤ r



Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r - k + 1$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19			

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step k-1 ≥ 2. Case 2: k ≤ r

Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r - k + 1$

k' =β k

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19			

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

Case 2b: $Z(k') \geq |\beta|$ $\quad |\beta| = r - k + 1$

k' =β

k

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19			

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r - k + 1$

↳ P[k...r]

↳ = P[k'...k'+|β|-1]

k' = β

k

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19			

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r - k + 1$

↳ P[k...r] matches P[1...|β|].

↳ = P[k'...k'+|β|-1] =

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19			

k' = 2 = β

k = 20

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r - k + 1$

↳ P[k...r] matches P[1...|β|]. Maybe more?

k' =β

k

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19			

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r - k + 1$

↳ P[k...r] matches P[1...|β|]. Maybe more?

↳ keep matching P[r+1...] w/ P[|β|+1...]

k' =β

k

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19			

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$

Z-box contains k

↳ matches prefix

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r-k+1$

↳ P[k...r] matches P[1...|β|]. Maybe more?

↳ keep matching P[r+1...] w/ P[|β|+1...]

β = suffix of Z-box ↳ starting at k

$k' = k-l+1$

$l = k$

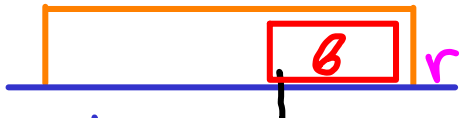
$Z_k = |\beta| + ?$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2			
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20		

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$



Z-box contains k

↳ matches prefix

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r-k+1$

↳ P[k...r] matches P[1...|β|]. Maybe more?

↳ keep matching P[r+1...] w/ P[|β|+1...]

β = suffix of Z-box ↳ starting at k

$k' = k-l+1$

$l = k$

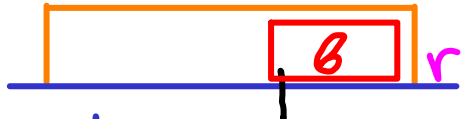
$Z_k = |\beta| + Q$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3		
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20		

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$



Z-box contains k

↳ matches prefix

β = suffix of Z-box ↳ starting at k

$k' = k - l + 1$

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r - k + 1$

↳ P[k...r] matches P[1...|β|]. Maybe more?

↳ keep matching P[r+1...] w/ P[|β|+1...]

k' =β

k

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3		
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20			
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20		

$l = k$

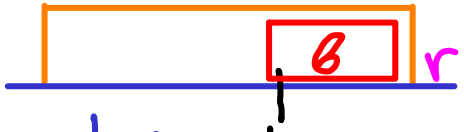
$Z_k = |\beta| + Q$

- Time = O(1) + O(Q).

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$



Z-box contains k
↳ matches prefix

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r-k+1$
↳ P[k...r] matches P[1...|β|]. Maybe more?

↳ keep matching P[r+1...] w/ P[|β|+1...]

β = suffix of Z-box
↳ starting at k

$k' = k-l+1$

$l = k$

$Z_k = |\beta| + Q$

- Time = O(1) + O(Q).
- r advances by Q.

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3		
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22		
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20		

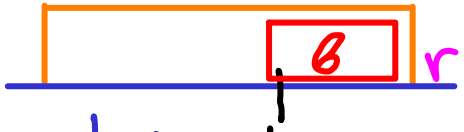
k' = β

k

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$



Z-box contains k

↳ matches prefix

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r-k+1$

↳ P[k...r] matches P[1...|β|]. Maybe more?

↳keep matching P[r+1...] w/ P[|β|+1...]

β = suffix of Z-box ↳starting at k

$k' = k-l+1$

$l = k$

$Z_k = |\beta| + Q$

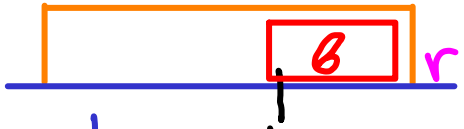
i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3		
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22		
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20		

- Time = O(1) + O(Q).
- r advances by Q.
- $r \leq n$, so...

Phase 1: Compute all Z_i of P in O(n) time

↳ get each Z_k in O(1) using only r_{k-1} & l_{k-1}

Assume we know r & l at step $k-1 \geq 2$. Case 2: $k \leq r$



Z-box contains k

↳ matches prefix

Case 2b: $Z(k') \geq |\beta|$ $|\beta| = r-k+1$

↳ P[k...r] matches P[1...|β|]. Maybe more?

↳keep matching P[r+1...] w/ P[|β|+1...]

β = suffix of Z-box ↳starting at k

$k' = k-l+1$

$l = k$

$Z_k = |\beta| + Q$

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3		
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22		
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20		

- Time = O(1) + O(Q).
- r advances by Q.
- r ≤ n, so...

ALL Case 2: O(n)

Completed Phase 1: Compute all Z_i of P in O(n) time & O(n) extra space

Z_i = | longest string starting at P(i) & matching prefix of P | $(i > 1)$

i:		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:		A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Completed Phase 1: Compute all Z_i of P in O(n) time & O(n) extra space

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Phase 2: Run Phase 1 algo on P$T ($ not in P or T)

AND$ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Completed Phase 1: Compute all Z_i of P in $O(n)$ time & $O(n)$ extra space

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Phase 2: Run Phase 1 algo on P$T ($ not in P or T)

- match iff $Z_i = n$ (i.e. if $r-l+1=n$)

AND$ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY

Z: / • • 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 3 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 3 0 0 0 1 0 0 0 3 0 0 1 0 0 0 0 0 0 0

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Completed Phase 1: Compute all Z_i of P in $O(n)$ time & $O(n)$ extra space

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Phase 2: Run Phase 1 algo on P$T ($ not in P or T)

↳ Max $Z_i = n$ → also $Z_{k'} \leq n$

- match iff $Z_i = n$ (i.e. if $r-l+1=n$)

AND$ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY

Z: / • • 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 3 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 3 0 0 0 0 1 0 0 0 3 0 0 1 0 0 0 0 0 0 0

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Completed Phase 1: Compute all Z_i of P in $O(n)$ time & $O(n)$ extra space

Z_i = |longest string starting at P(i) & matching prefix of P| $(i>1)$

Phase 2: Run Phase 1 algo on P$T ($ not in P or T)

↳ Max $Z_i = n$ → also $Z_{k'} \leq n$

- match iff $Z_i = n$ (i.e. if $r-l+1=n$) // only need to store $Z_{1\ldots n}$: no extra space

AND$ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY

Z: / • • 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 3 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 3 0 0 0 1 0 0 0 3 0 0 1 0 0 0 0 0 0 0

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

Completed Phase 1: Compute all Z_i of P in $O(n)$ time & $O(n)$ extra space

Z_i = | longest string starting at P(i) & matching prefix of P | $(i > 1)$

Phase 2: Run Phase 1 algo on P$T ($ not in P or T)

↳ Max $Z_i = n$ → also $Z_{k'} \leq n$

- match iff $Z_i = n$ (i.e. if $r - l + 1 = n$) // only need to store $Z_{1 \ldots n}$: no extra space

AND$ILOVEALGORITHMSANDDATASTRUCTURESANDGRAPHSANDALLTHEORY

Z: / • • 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 3 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 3 0 0 0 1 0 0 0 3 0 0 1 0 0 0 0 0 0 0

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P:	A	A	B	A	A	B	C	A	X	A	A	B	A	A	B	C	Y	A	A	A	A	B
Z	/	1	0	3	1	0	0	1	0	7	1	0	3	1	0	0	0	2	2	3	1	0
r	/	2	2	6	6	6	6	8	8	16	16	16	16	16	16	16	16	19	20	22	22	22
l	/	2	2	4	4	4	4	8	8	10	10	10	10	10	10	10	10	18	19	20	20	20

TOTAL TIME:

$\Theta(n+m) = \Theta(m)$

total extra space $= \Theta(n)$