Advanced Algorithms – COMS31900

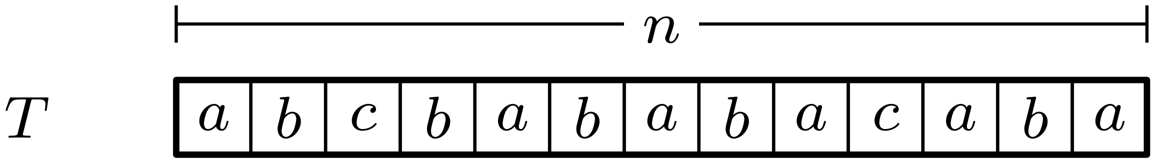
Pattern Matching part two

Suffix Arrays

Benjamin Sach

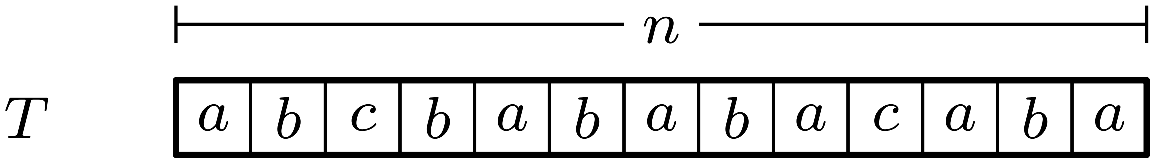
Text indexing

Preprocess a text string T (length n) to answer pattern matching queries. . .



Text indexing

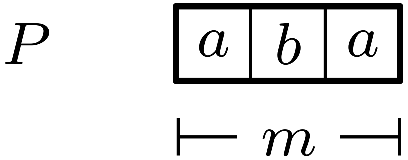
Preprocess a text string T (length n) to answer pattern matching queries...



n

T	a	b	c	b	a	b	a	b	a	c	a	b	a

After preprocessing, a **query** is a pattern P (length m),

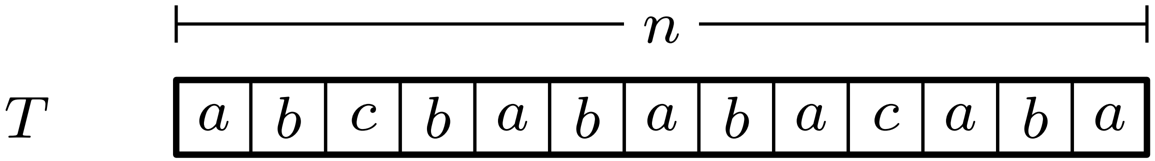


P	a	b	a

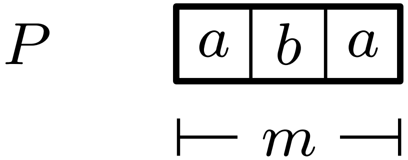
m

Text indexing

Preprocess a text string T (length n) to answer pattern matching queries...



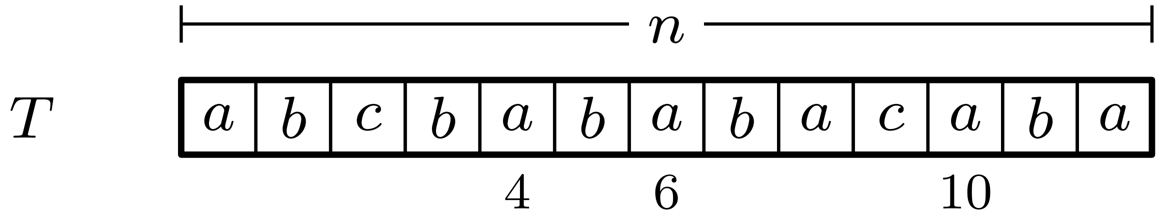
After preprocessing, a **query** is a pattern P (length m),



the output is a list of all matches in T.

Text indexing

Preprocess a text string T (length n) to answer pattern matching queries. . .

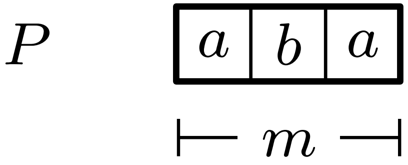


n

T | a | b | c | b | a | b | a | b | a | c | a | b | a |

4 6 10

After preprocessing, a **query** is a pattern P (length m),



P | a | b | a |

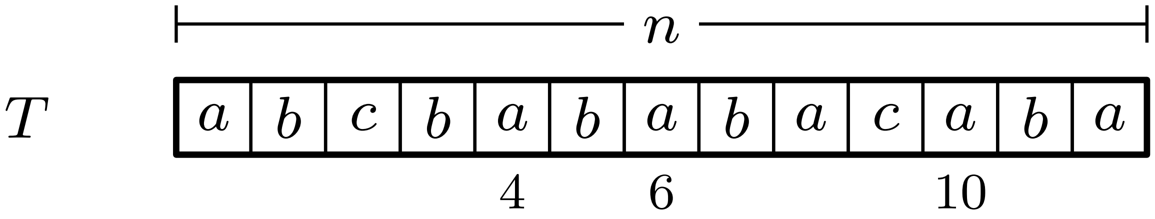
m

the output is a list of all matches in T.

e.g. $4, 6, 10$

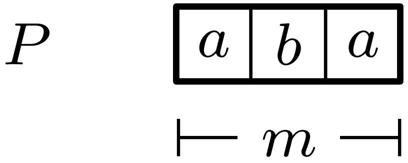
Text indexing

Preprocess a text string T (length n) to answer pattern matching queries. . .



After preprocessing, a **query** is a pattern P (length m),

the output is a list of all matches in T.

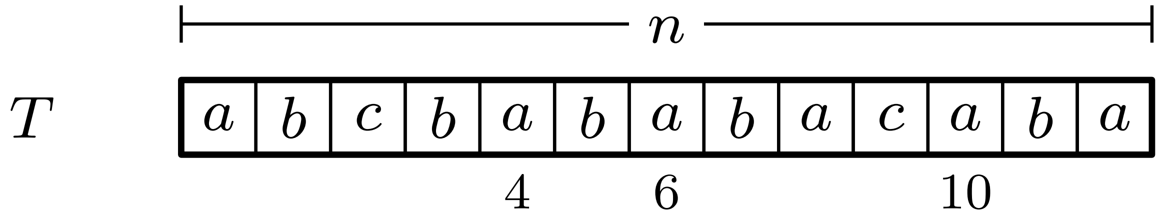


e.g. $4, 6, 10$

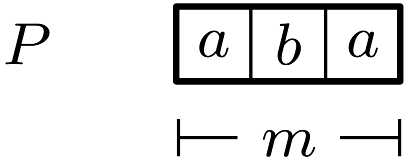
- Last lecture we saw the that **text indexing** problem can be solved using a suffix tree which uses $O(n)$ space *(when it's stored compacted)*

Text indexing

Preprocess a text string T (length n) to answer pattern matching queries. . .



After preprocessing, a **query** is a pattern P (length m),



the output is a list of all matches in T.

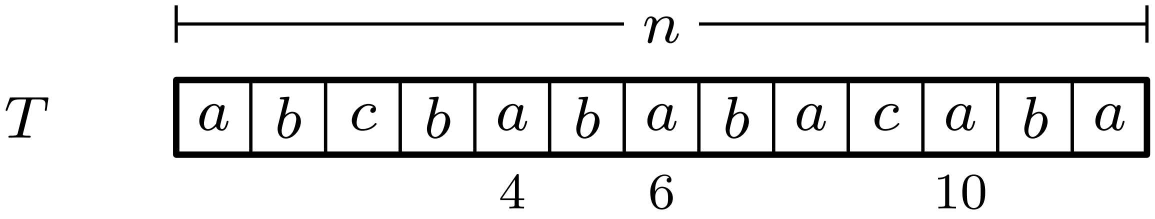
e.g. $4, 6, 10$

- Last lecture we saw the that **text indexing** problem can be solved using a suffix tree which uses $O(n)$ space *(when it's stored compacted)*

- Queries take $O(m + \text{occ})$ time when the alphabet size is constant
  - occ is the number of occurences (matches)

Text indexing

Preprocess a text string T (length n) to answer pattern matching queries...



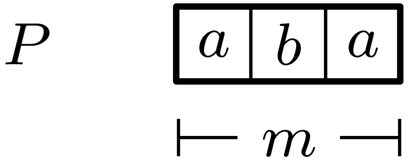
n

T | a | b | c | b | a | b | a | b | a | c | a | b | a

4 6 10

After preprocessing, a **query** is a pattern P (length m),

the output is a list of all matches in T.



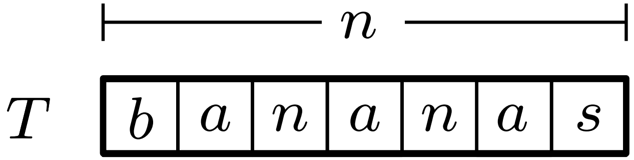
P | a | b | a

m

e.g. $4, 6, 10$

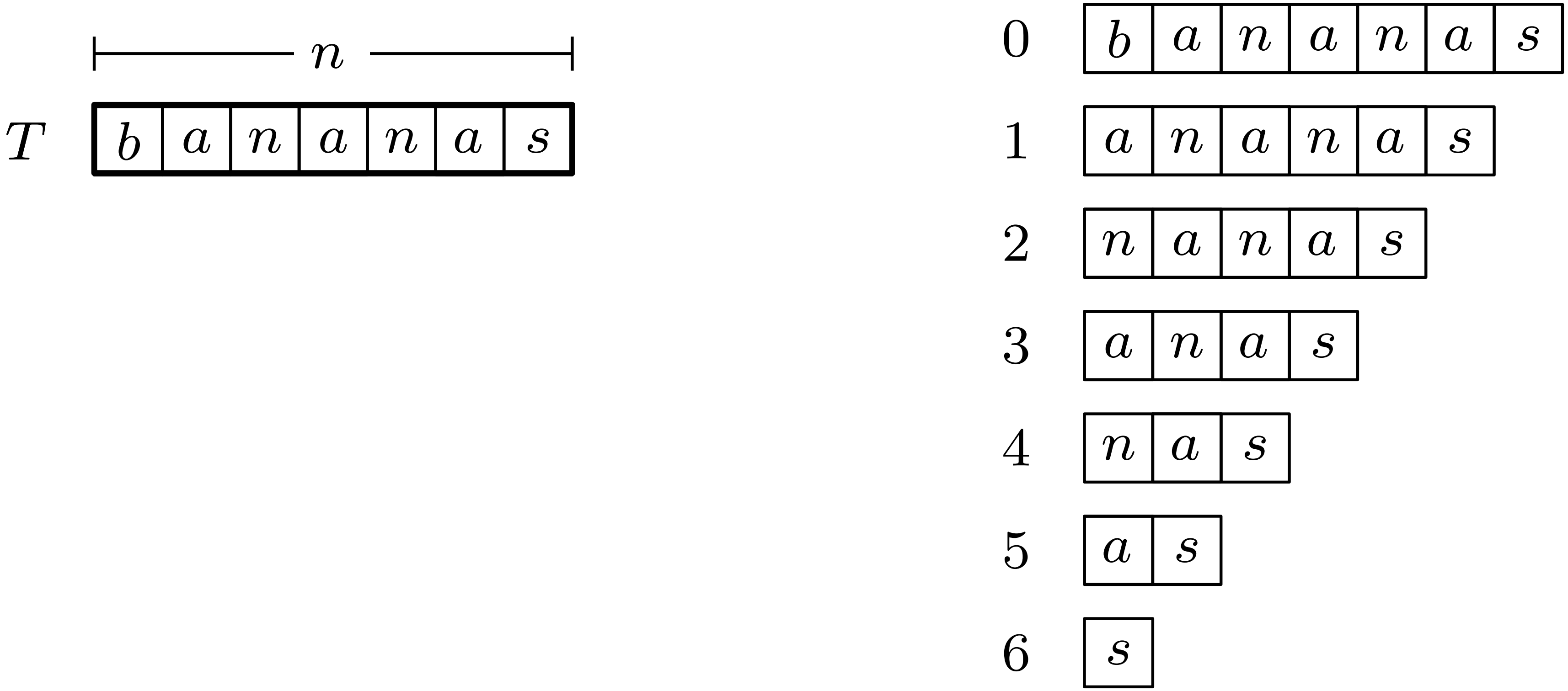
- Last lecture we saw the that **text indexing** problem can be solved using a suffix tree which uses $O(n)$ space *(when it's stored compacted)*
- Queries take $O(m + \text{occ})$ time when the alphabet size is constant
  - occ is the number of occurences (matches)
- Suffix trees can be constructed in $O(n)$ time (but we only saw how to achieve $O(n^2)$ time)

The suffix array



T	b	a	n	a	n	a	s

The suffix array



0 b a n a n a s

1 a n a n a s

2 n a n a s

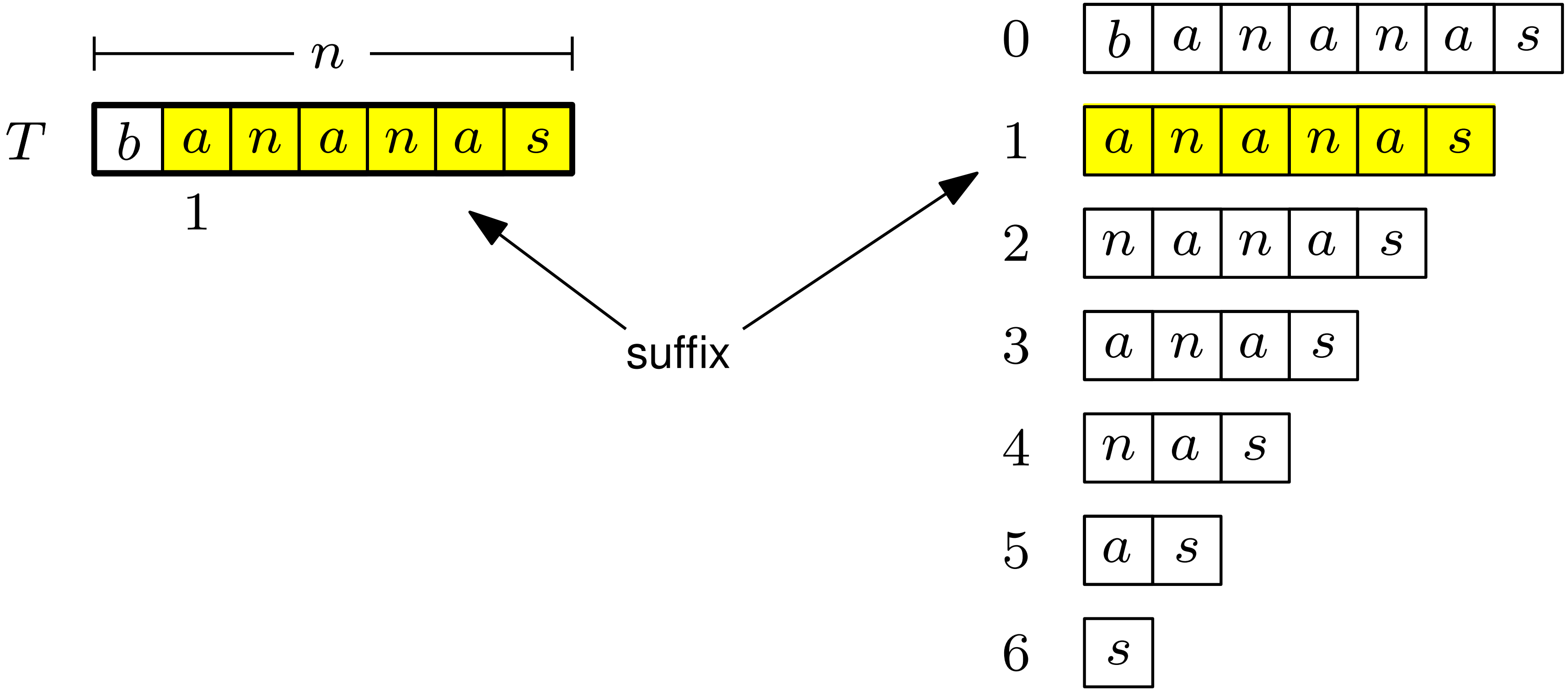
3 a n a s

4 n a s

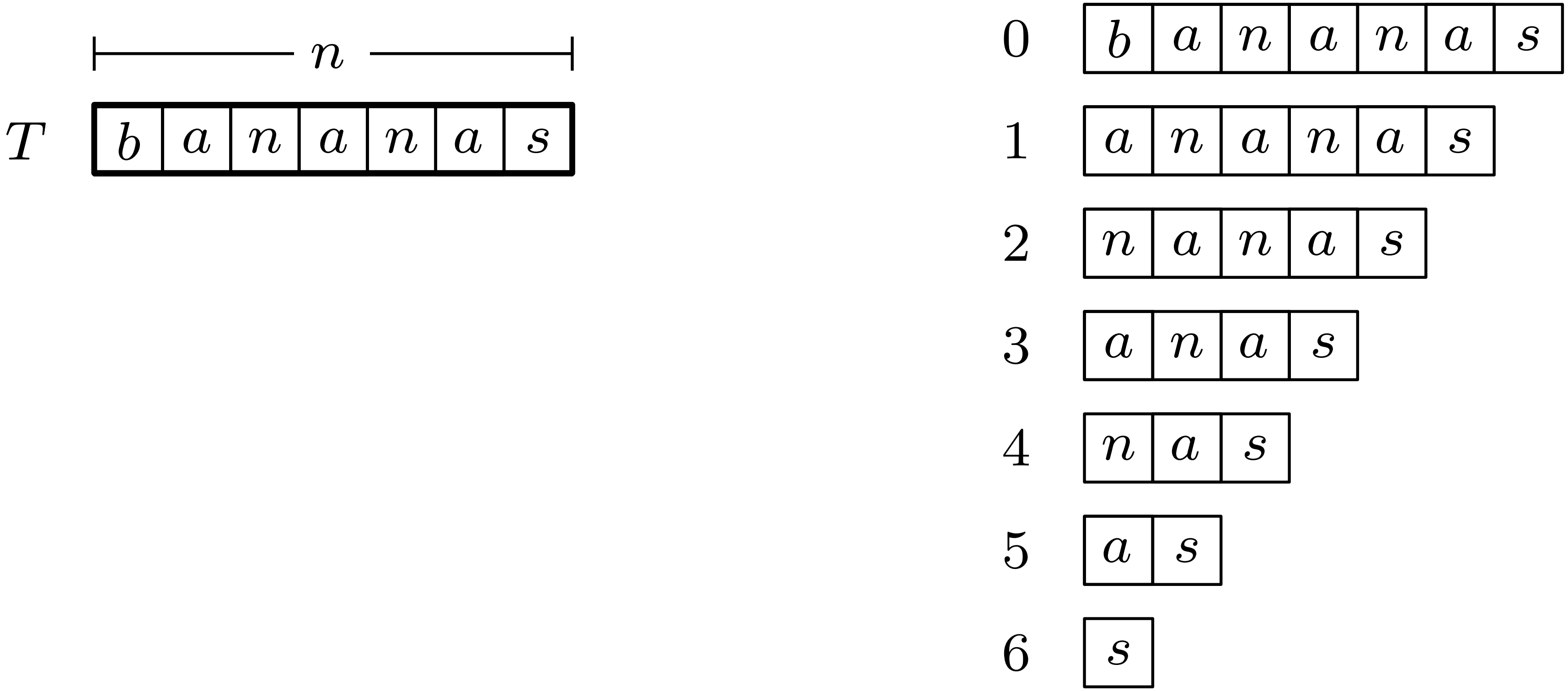
5 a s

6 s

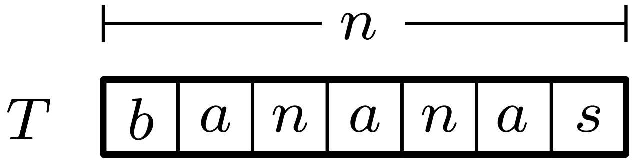
The suffix array



The suffix array



The suffix array



Sort the suffixes lexicographically

0	$b\ a\ n\ a\ n\ a\ s$
1	$a\ n\ a\ n\ a\ s$
2	$n\ a\ n\ a\ s$
3	$a\ n\ a\ s$
4	$n\ a\ s$
5	$a\ s$
6	s

The suffix array

$$T \quad \overbrace{\boxed{b}\boxed{a}\boxed{n}\boxed{a}\boxed{n}\boxed{a}\boxed{s}}^{n}$$

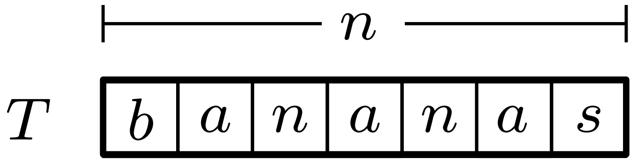
0	b	a	n	a	n	a	s
1	a	n	a	n	a	s	
2	n	a	n	a	s		
3	a	n	a	s			
4	n	a	s				
5	a	s					
6	s						

Sort the suffixes lexicographically

- The symbols themselves must have an order

throughout we will use alphabetical order

The suffix array



0	b	a	n	a	n	a	s
1	a	n	a	n	a	s	
2	n	a	n	a	s		
3	a	n	a	s			
4	n	a	s				
5	a	s					
6	s						

Sort the suffixes

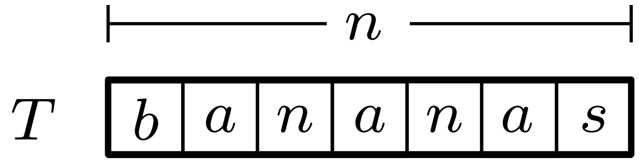
lexicographically

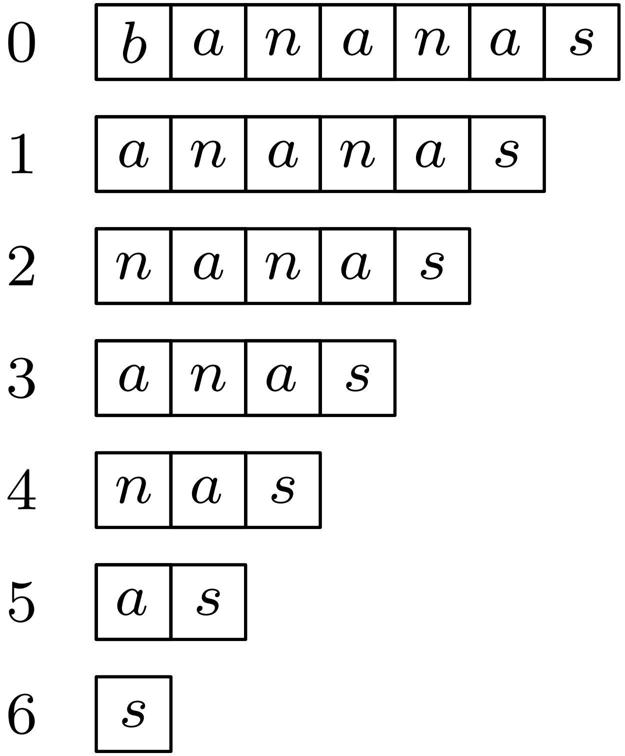
- The symbols themselves must have an order

  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:

The suffix array





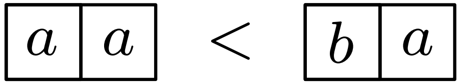
Sort the suffixes

lexicographically

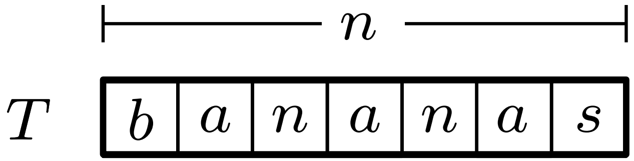
- The symbols themselves must have an order

  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:

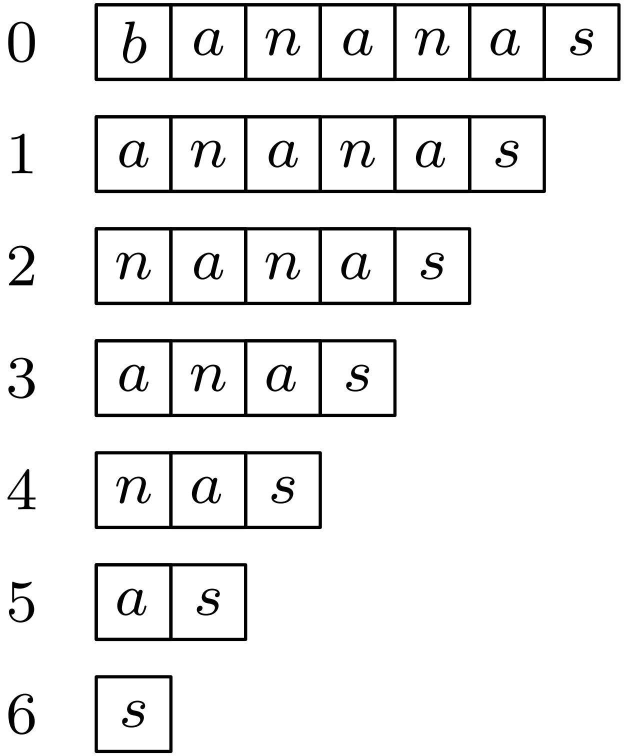


The suffix array



n

T | b | a | n | a | n | a | s |



0 $b\ a\ n\ a\ n\ a\ s$

1 $a\ n\ a\ n\ a\ s$

2 $n\ a\ n\ a\ s$

3 $a\ n\ a\ s$

4 $n\ a\ s$

5 $a\ s$

6 s

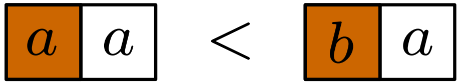
Sort the suffixes

lexicographically

- The symbols themselves must have an order

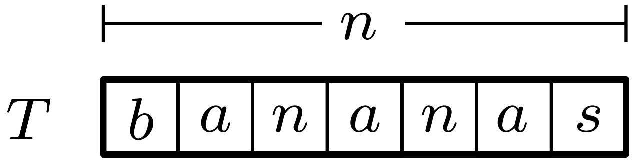
  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:



$a\ a < b\ a$

The suffix array



0	b	a	n	a	n	a	s
1	a	n	a	n	a	s	
2	n	a	n	a	s		
3	a	n	a	s			
4	n	a	s				
5	a	s					
6	s						

Sort the suffixes

lexicographically

- The symbols themselves must have an order

  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:

$$\boxed{a}\boxed{a} < \boxed{b}\boxed{a}$$

The suffix array

$$T \quad \overbrace{\boxed{b}\boxed{a}\boxed{n}\boxed{a}\boxed{n}\boxed{a}\boxed{s}}^{n}$$

Sort the suffixes

lexicographically

- The symbols themselves must have an order

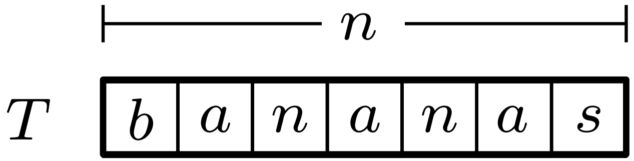
throughout we will use alphabetical order

0	b	a	n	a	n	a	s
1	a	n	a	n	a	s	
2	n	a	n	a	s		
3	a	n	a	s			
4	n	a	s				
5	a	s					
6	s						

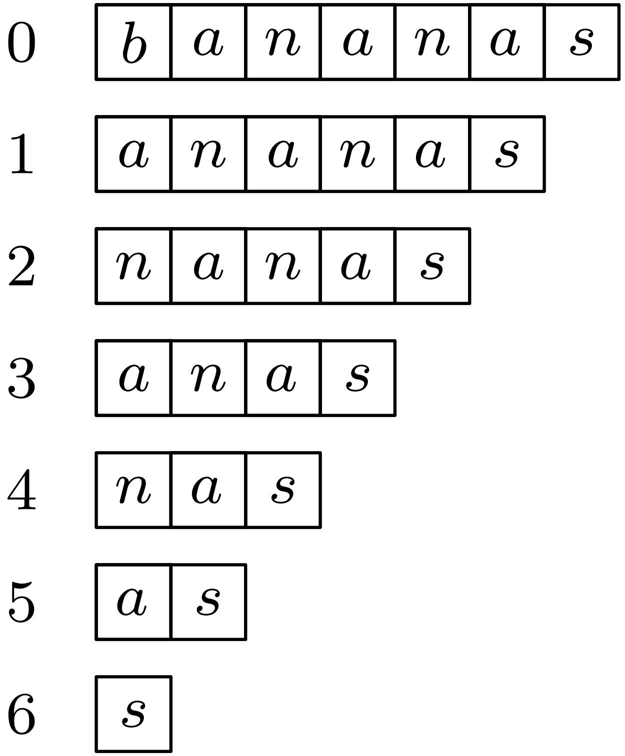
In lexicographical ordering we sort strings based on the first symbol that differs:

$$\boxed{a}\boxed{a} < \boxed{b}\boxed{a} < \boxed{b}\boxed{c}$$

The suffix array



Sort the suffixes lexicographically



0 $bananas$

1 $ananas$

2 $nanas$

3 $anas$

4 nas

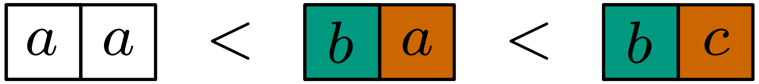
5 as

6 s

- The symbols themselves must have an order

  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:



$aa < ba < bc$

The suffix array

n

T	b	a	n	a	n	a	s

Sort the suffixes

lexicographically

0	b	a	n	a	n	a	s
1	a	n	a	n	a	s	
2	n	a	n	a	s		
3	a	n	a	s			
4	n	a	s				
5	a	s					
6	s						

- The symbols themselves must have an order

  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:

$$\boxed{a\,|\,a} < \boxed{b\,|\,a} < \boxed{b\,|\,c}$$

The suffix array

$\vdash\!\longleftarrow\; n \;\longrightarrow\!\dashv$

T	b	a	n	a	n	a	s

0	b	a	n	a	n	a	s
1	a	n	a	n	a	s	
2	n	a	n	a	s		
3	a	n	a	s			
4	n	a	s				
5	a	s					
6	s						

Sort the suffixes lexicographically

- The symbols themselves must have an order

  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:

$$\boxed{a}\boxed{a} < \boxed{b}\boxed{a} < \boxed{b}\boxed{c} < \boxed{b}\boxed{c}\boxed{a}$$

The suffix array

$\overbrace{\hspace{6em}}^{n}$

T	b	a	n	a	n	a	s

0	b	a	n	a	n	a	s
1	a	n	a	n	a	s	
2	n	a	n	a	s		
3	a	n	a	s			
4	n	a	s				
5	a	s					
6	s						

Sort the suffixes lexicographically

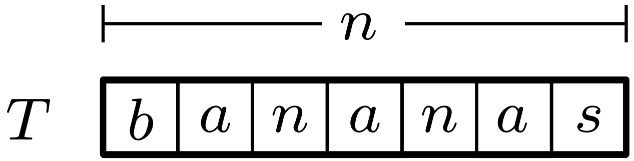
- The symbols themselves must have an order

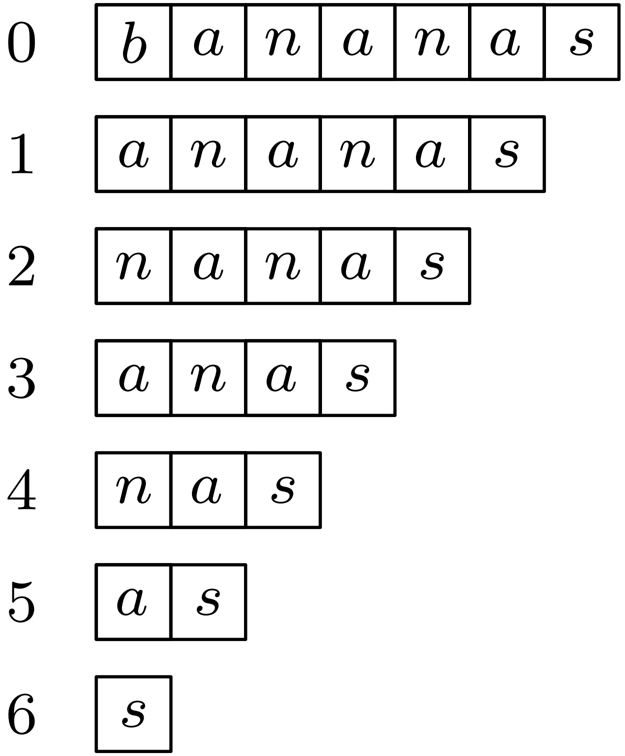
  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:

$$\boxed{a}\boxed{a} < \boxed{b}\boxed{a} < \boxed{b}\boxed{c} < \boxed{b}\boxed{c}\boxed{a}$$

The suffix array





0	b	a	n	a	n	a	s
1	a	n	a	n	a	s	
2	n	a	n	a	s		
3	a	n	a	s			
4	n	a	s				
5	a	s					
6	s						

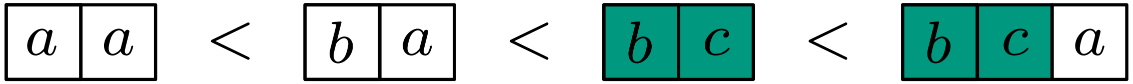
Sort the suffixes

lexicographically

- The symbols themselves must have an order

  throughout we will use alphabetical order

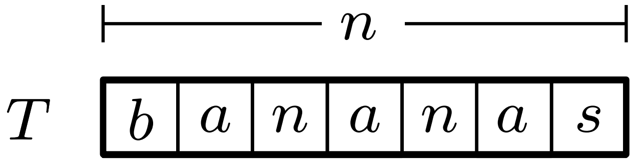
In lexicographical ordering we sort strings based on the first symbol that differs:



$a\,a < b\,a < b\,c < b\,c\,a$

(in a 'tie', the shorter string is smaller)

The suffix array



Sort the suffixes

lexicographically

0 $b\,a\,n\,a\,n\,a\,s$

1 $a\,n\,a\,n\,a\,s$

2 $n\,a\,n\,a\,s$

3 $a\,n\,a\,s$

4 $n\,a\,s$

5 $a\,s$

6 s

- The symbols themselves must have an order

  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:

$\boxed{a\,a} < \boxed{b\,a} < \boxed{b\,c} < \boxed{b\,c\,a}$

(in a ‘tie’, the shorter string is smaller)

The suffix array

$$T \quad \overset{\longleftarrow\; n \;\longrightarrow}{\begin{array}{|c|c|c|c|c|c|c|}\hline b & a & n & a & n & a & s \\ \hline \end{array}}$$

Sort the suffixes lexicographically

- The symbols themselves must have an order

  throughout we will use alphabetical order

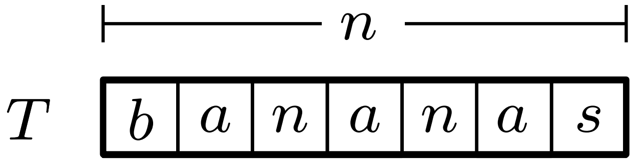
1	a	n	a	n	a	s	
3	a	n	a	s			
5	a	s					
0	b	a	n	a	n	a	s
2	n	a	n	a	s		
4	n	a	s				
6	s						

In lexicographical ordering we sort strings based on the first symbol that differs:

$$\begin{array}{|c|c|}\hline a & a \\ \hline \end{array} < \begin{array}{|c|c|}\hline b & a \\ \hline \end{array} < \begin{array}{|c|c|}\hline b & c \\ \hline \end{array} < \begin{array}{|c|c|c|}\hline b & c & a \\ \hline \end{array}$$

(in a 'tie', the shorter string is smaller)

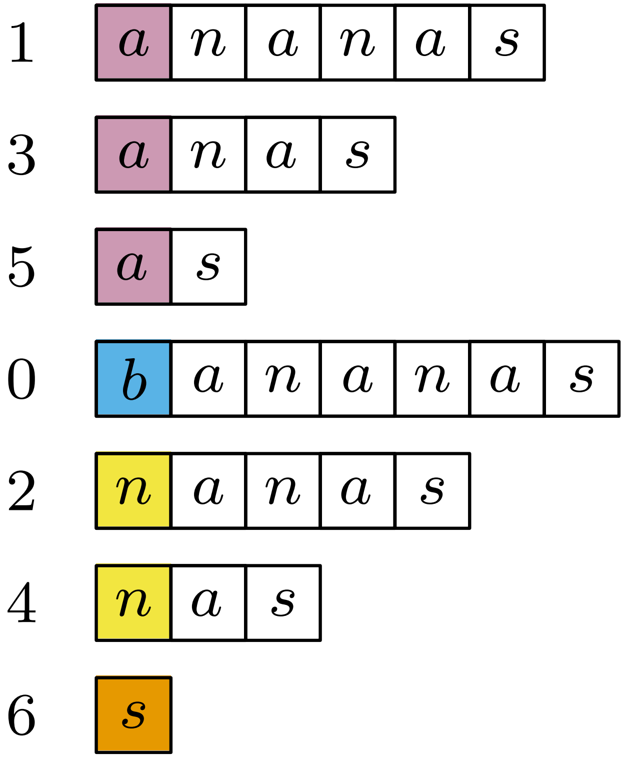
The suffix array



n

T $b\ a\ n\ a\ n\ a\ s$

Sort the suffixes lexicographically



1 $a\ n\ a\ n\ a\ s$

3 $a\ n\ a\ s$

5 $a\ s$

0 $b\ a\ n\ a\ n\ a\ s$

2 $n\ a\ n\ a\ s$

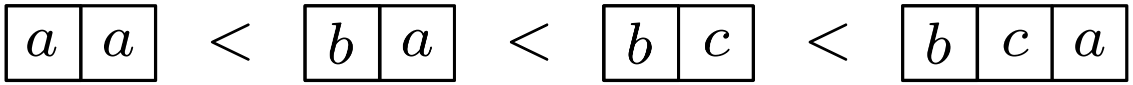
4 $n\ a\ s$

6 s

- The symbols themselves must have an order

  throughout we will use alphabetical order

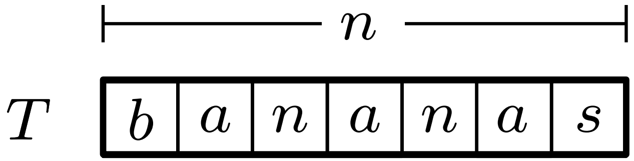
In lexicographical ordering we sort strings based on the first symbol that differs:



$a\ a < b\ a < b\ c < b\ c\ a$

(in a ‘tie’, the shorter string is smaller)

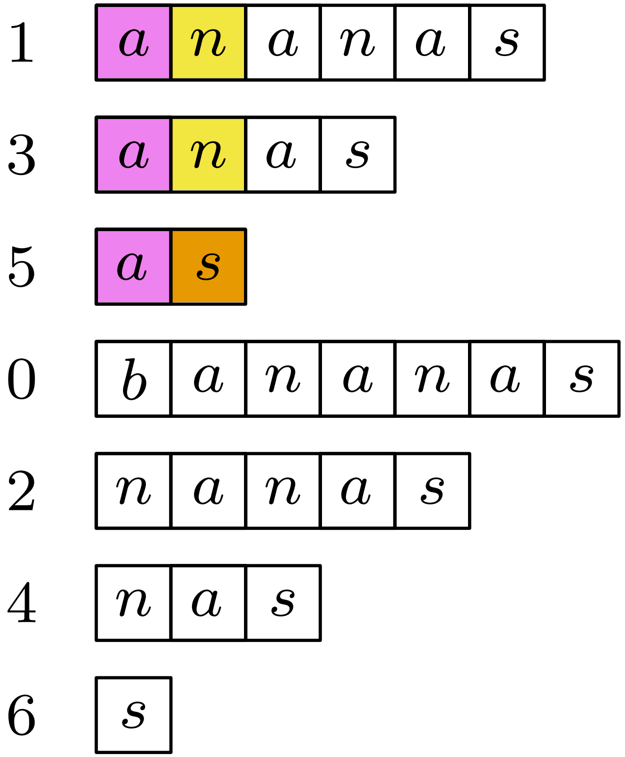
The suffix array



n

T $b\ a\ n\ a\ n\ a\ s$

Sort the suffixes lexicographically



1 $a\ n\ a\ n\ a\ s$

3 $a\ n\ a\ s$

5 $a\ s$

0 $b\ a\ n\ a\ n\ a\ s$

2 $n\ a\ n\ a\ s$

4 $n\ a\ s$

6 s

- The symbols themselves must have an order

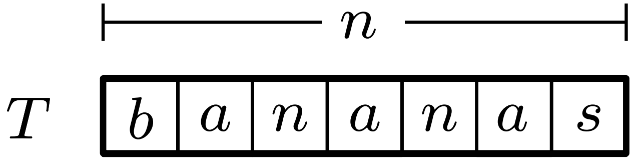
  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:

$$\boxed{a\,|\,a} < \boxed{b\,|\,a} < \boxed{b\,|\,c} < \boxed{b\,|\,c\,|\,a}$$

(in a ‘tie’, the shorter string is smaller)

The suffix array



Sort the suffixes lexicographically

1	a n a n a s
3	a n a s
5	a s
0	b a n a n a s
2	n a n a s
4	n a s
6	s

- The symbols themselves must have an order

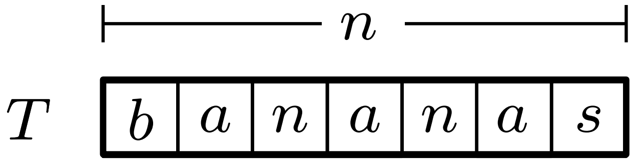
  throughout we will use alphabetical order

In lexicographical ordering we sort strings based on the first symbol that differs:

$$\boxed{a\,|\,a} < \boxed{b\,|\,a} < \boxed{b\,|\,c} < \boxed{b\,|\,c\,|\,a}$$

(in a 'tie', the shorter string is smaller)

The suffix array



Sort the suffixes lexicographically

1	a	n	a	n	a	s	
3	a	n	a	s			
5	a	s					
0	b	a	n	a	n	a	s
2	n	a	n	a	s		
4	n	a	s				
6	s						

- The symbols themselves must have an order

throughout we will use alphabetical order

just a fancy name for the order the strings would appear in the dictionary

In lexicographical ordering we sort strings based on the first symbol that differs:

$$\boxed{a\,|\,a} < \boxed{b\,|\,a} < \boxed{b\,|\,c} < \boxed{b\,|\,c\,|\,a}$$

(in a ‘tie’, the shorter string is smaller)

The suffix array

n

T	b	a	n	a	n	a	s

Sort the suffixes lexicographically

1	a	n	a	n	a	s	
3	a	n	a	s			
5	a	s					
0	b	a	n	a	n	a	s
2	n	a	n	a	s		
4	n	a	s				
6	s						

- The symbols themselves must have an order

throughout we will use alphabetical order

just a fancy name for the order the strings would appear in the dictionary

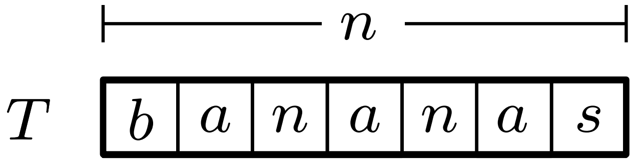
In lexicographical ordering we sort strings based on the first symbol that differs:

$$aa < ba < bc < bca$$

(in a 'tie', the shorter string is smaller)

If the symbols don't have a natural order, we use their binary representation in memory

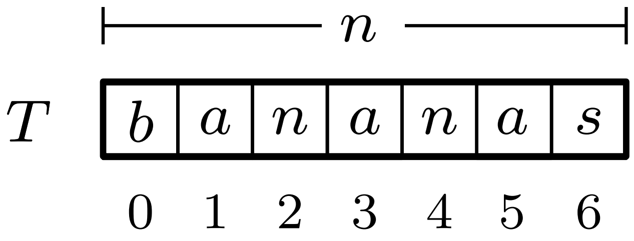
The suffix array



Sort the suffixes lexicographically

1	a	n	a	n	a	s	
3	a	n	a	s			
5	a	s					
0	b	a	n	a	n	a	s
2	n	a	n	a	s		
4	n	a	s				
6	s						

The suffix array

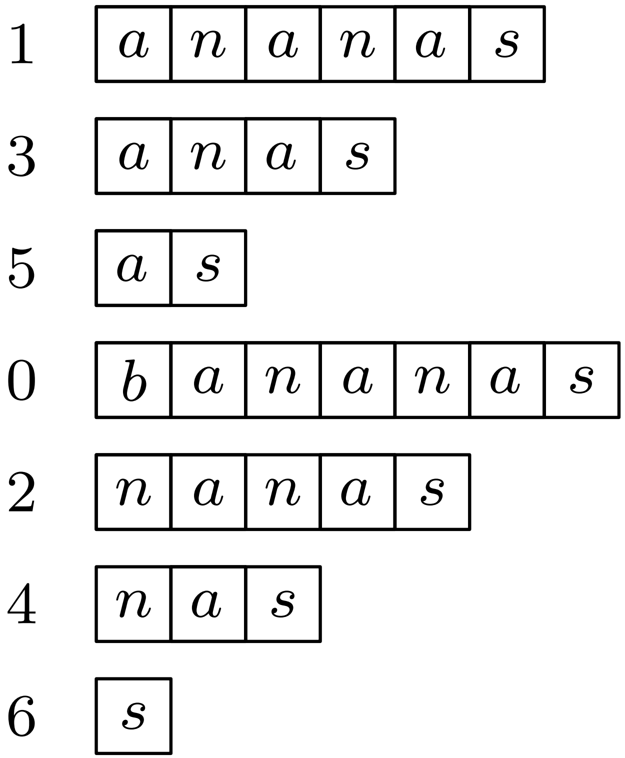


n

T b a n a n a s

0 1 2 3 4 5 6

Sort the suffixes lexicographically



1 a n a n a s

3 a n a s

5 a s

0 b a n a n a s

2 n a n a s

4 n a s

6 s

The suffix array

n

T	b	a	n	a	n	a	s
	0	1	2	3	4	5	6

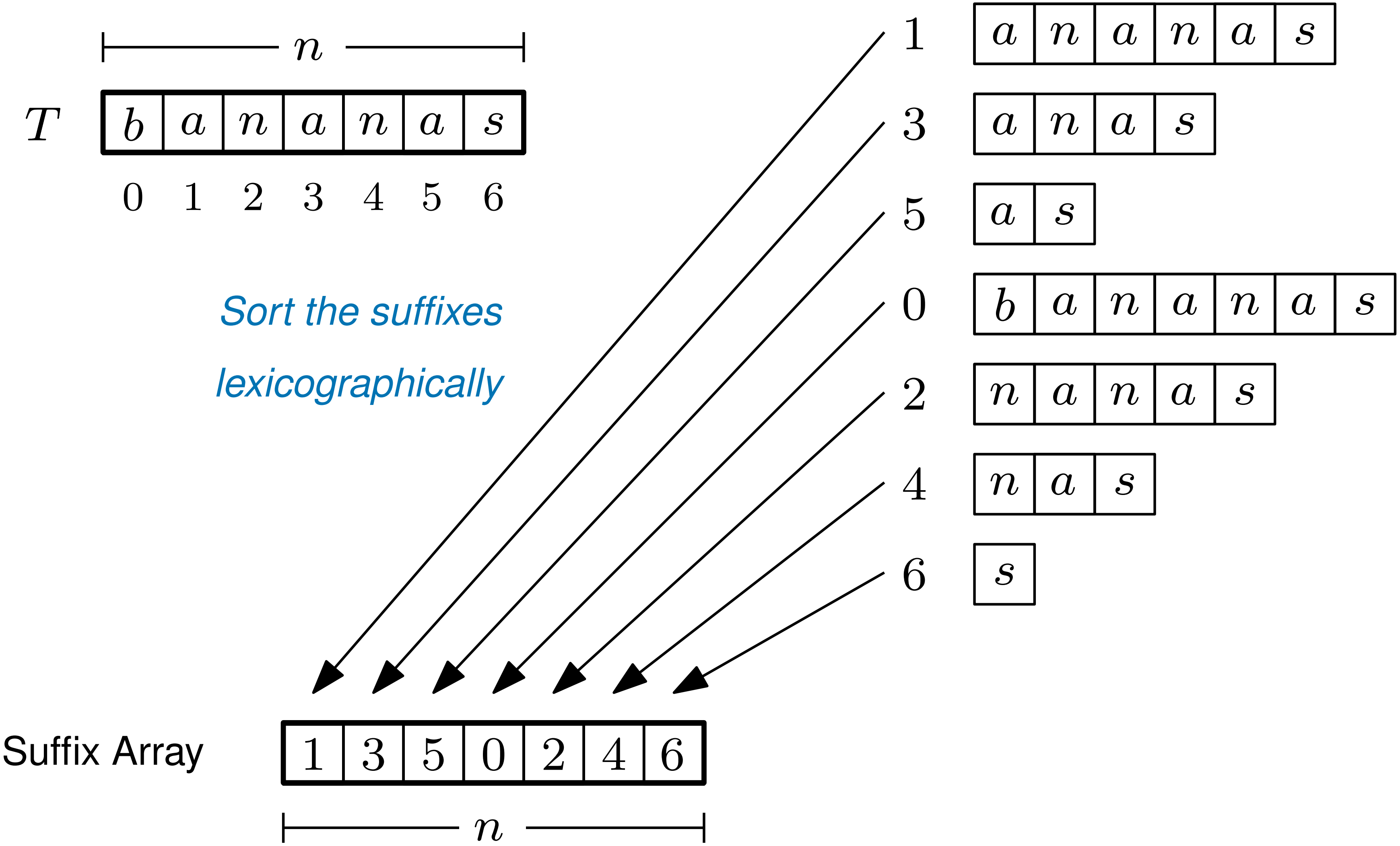
Sort the suffixes lexicographically

1	a	n	a	n	a	s	
3	a	n	a	s			
5	a	s					
0	b	a	n	a	n	a	s
2	n	a	n	a	s		
4	n	a	s				
6	s						

Suffix Array	1	3	5	0	2	4	6

n

The suffix array



n

T	b	a	n	a	n	a	s
	0	1	2	3	4	5	6

Sort the suffixes lexicographically

1	$a\ n\ a\ n\ a\ s$
3	$a\ n\ a\ s$
5	$a\ s$
0	$b\ a\ n\ a\ n\ a\ s$
2	$n\ a\ n\ a\ s$
4	$n\ a\ s$
6	s

Suffix Array	1	3	5	0	2	4	6

n

The suffix array

n

T	b	a	n	a	n	a	s
	0	1	2	3	4	5	6

Sort the suffixes lexicographically

1	a	n	a	n	a	s	
3	a	n	a	s			
5	a	s					
0	b	a	n	a	n	a	s
2	n	a	n	a	s		
4	n	a	s				
6	s						

Suffix Array	1	3	5	0	2	4	6

n

The suffix array

n

T	b	a	n	a	n	a	s
	0	1	2	3	4	5	6

Sort the suffixes lexicographically

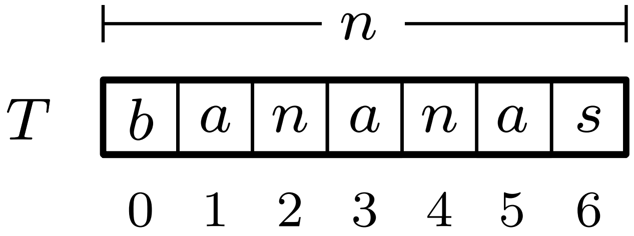
1	a	n	a	n	a	s	
3	a	n	a	s			
5	a	s					
0	b	a	n	a	n	a	s
2	n	a	n	a	s		
4	n	a	s				
6	s						

Suffix Array	1	3	5	0	2	4	6

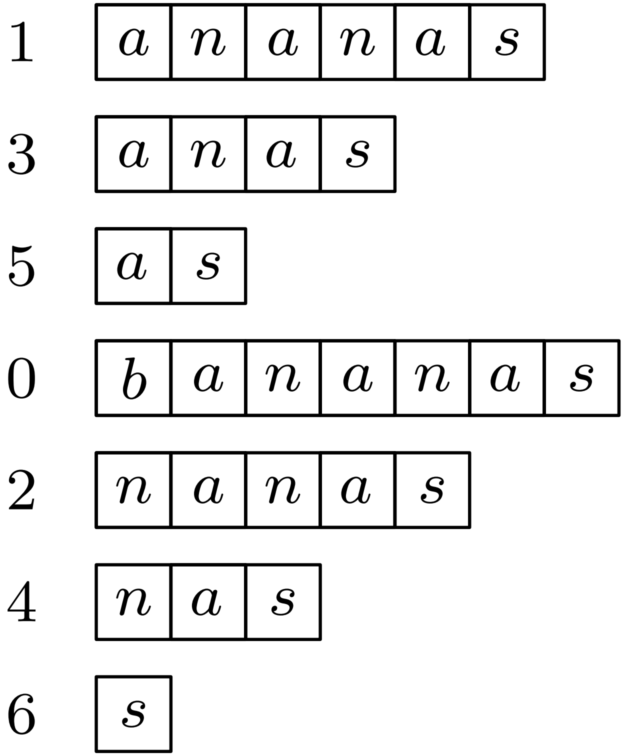
n

The suffix array is much smaller than the suffix tree (in terms of constants)

The suffix array

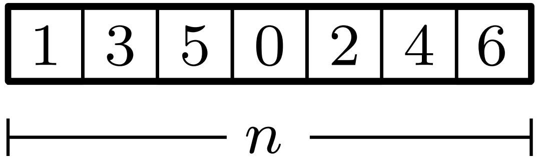


Sort the suffixes lexicographically



1	$a\ n\ a\ n\ a\ s$
3	$a\ n\ a\ s$
5	$a\ s$
0	$b\ a\ n\ a\ n\ a\ s$
2	$n\ a\ n\ a\ s$
4	$n\ a\ s$
6	s

Suffix Array



The suffix array is much smaller than the suffix tree (in terms of constants)

From Suffix Trees to Suffix Arrays

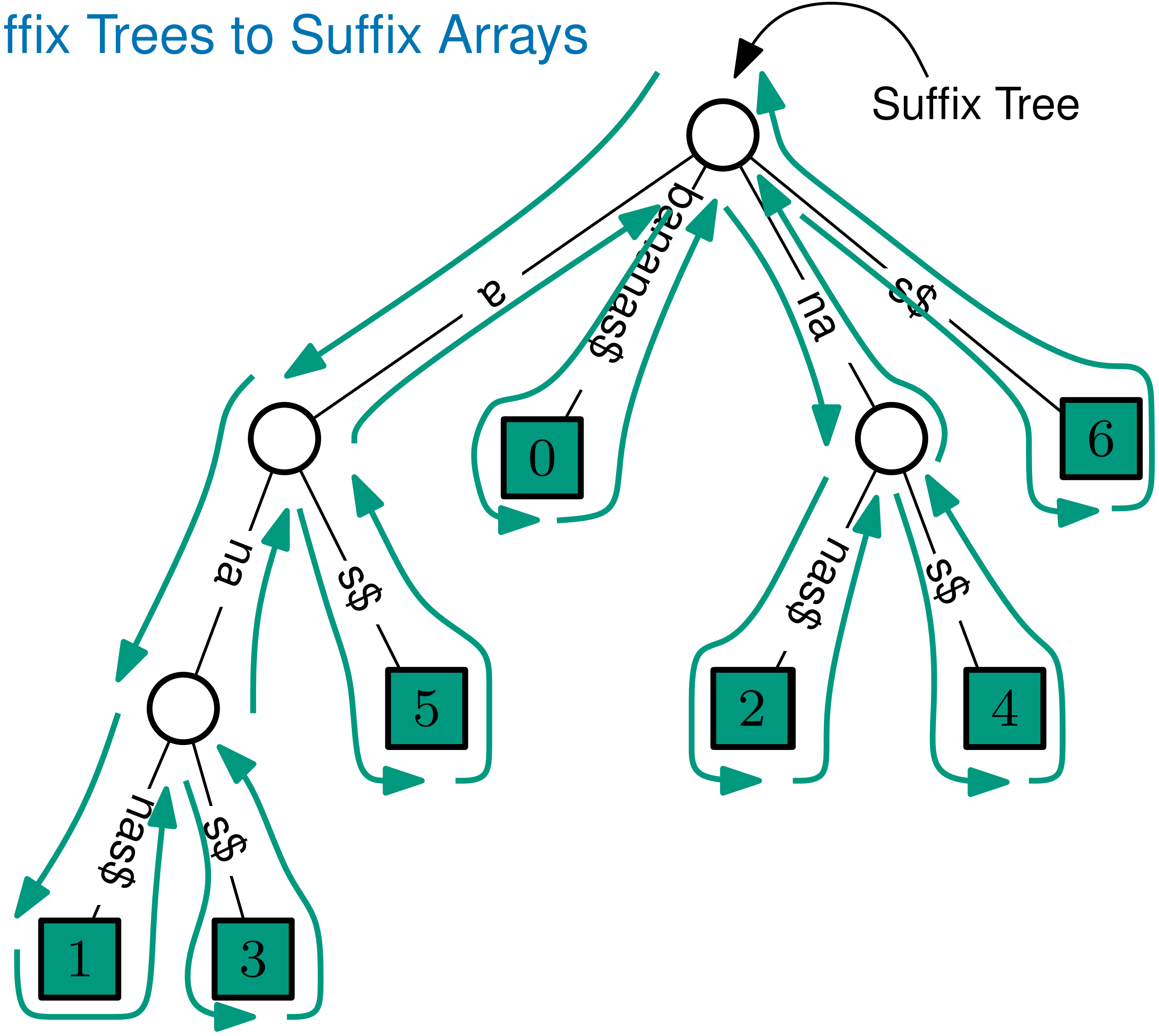
$\vdash\!\!-\!\!-\; n \;-\!\!-\!\!\dashv$

T	b	a	n	a	n	a	s
	0	1	2	3	4	5	6

Suffix Array

1	3	5	0	2	4	6

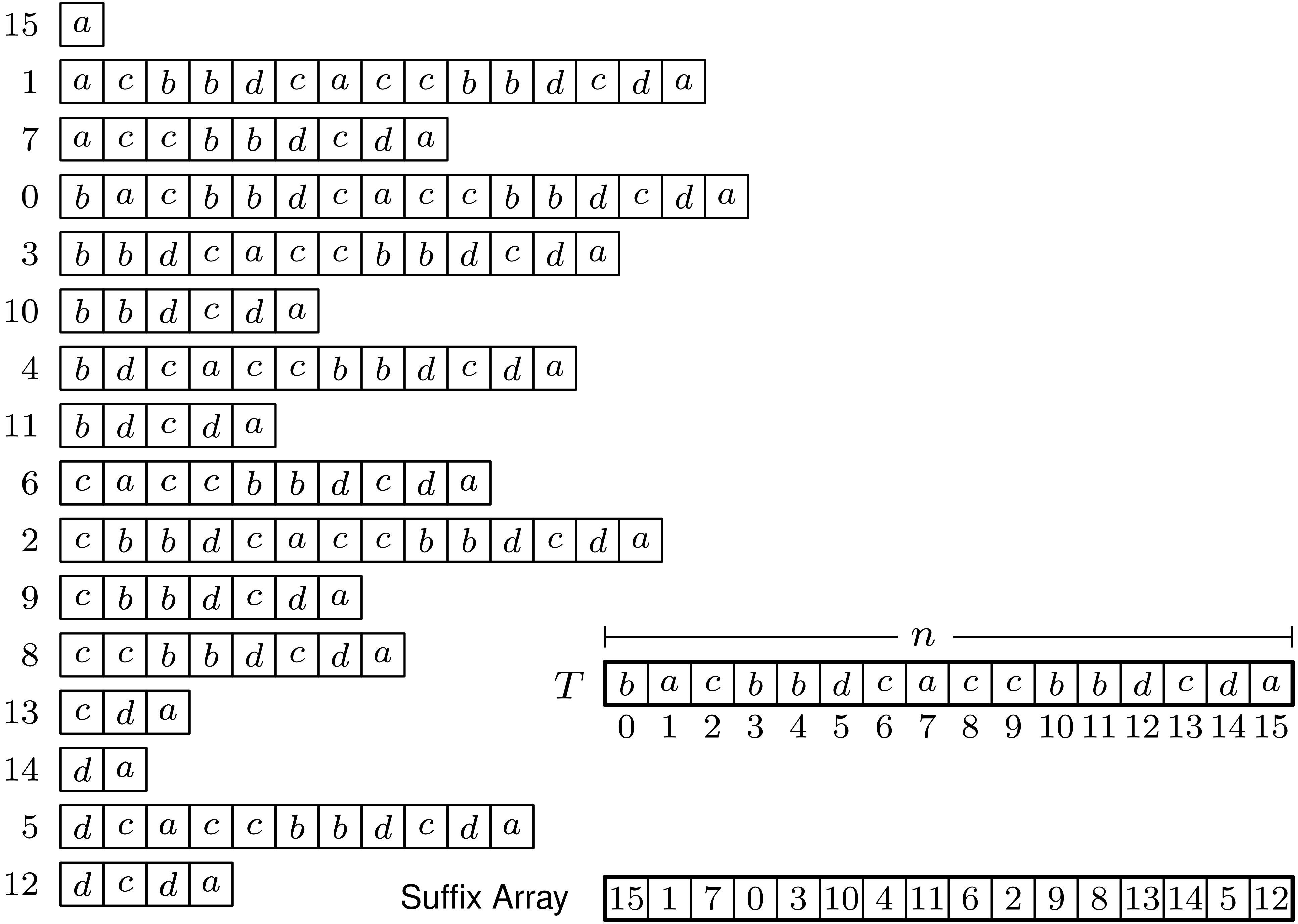
$\vdash\!\!-\!\!-\; n \;-\!\!-\!\!\dashv$



Recall that we can get get the Suffix Array from the Suffix Tree

using depth-first search in $O(n)$ time

Searching in the Suffix Array



15	a															
1	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a	
7	a	c	c	b	b	d	c	d	a							
0	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
3	b	b	d	c	a	c	c	b	b	d	c	d	a			
10	b	b	d	c	d	a										
4	b	d	c	a	c	c	b	b	d	c	d	a				
11	b	d	c	d	a											
6	c	a	c	c	b	b	d	c	d	a						
2	c	b	b	d	c	a	c	c	b	b	d	c	d	a		
9	c	b	b	d	c	d	a									
8	c	c	b	b	d	c	d	a								
13	c	d	a													
14	d	a														
5	d	c	a	c	c	b	b	d	c	d	a					
12	d	c	d	a												

n

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array

15 a
1 a c b b d c a c c b b d c d a
7 a c c b b d c d a
0 b a c b b d c a c c b b d c d a
3 b b d c a c c b b d c d a
10 b b d c d a
4 b d c a c c b b d c d a
11 b d c d a
6 c a c c b b d c d a
2 c b b d c a c c b b d c d a
9 c b b d c d a
8 c c b b d c d a
13 c d a
14 d a
5 d c a c c b b d c d a
12 d c d a

find P (length m)

c	b	b	d	c

T (length n)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array

15	a															
1	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a	
7	a	c	c	b	b	d	c	d	a							
0	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
3	b	b	d	c	a	c	c	b	b	d	c	d	a			
10	b	b	d	c	d	a										
4	b	d	c	a	c	c	b	b	d	c	d	a				
11	b	d	c	d	a											
6	c	a	c	c	b	b	d	c	d	a						
2	c	b	b	d	c	a	c	c	b	b	d	c	d	a		
9	c	b	b	d	c	d	a									
8	c	c	b	b	d	c	d	a								
13	c	d	a													
14	d	a														
5	d	c	a	c	c	b	b	d	c	d	a					
12	d	c	d	a												

find P (length m):

c	b	b	d	c

Key Idea:

Find an occurence of P

using binary search

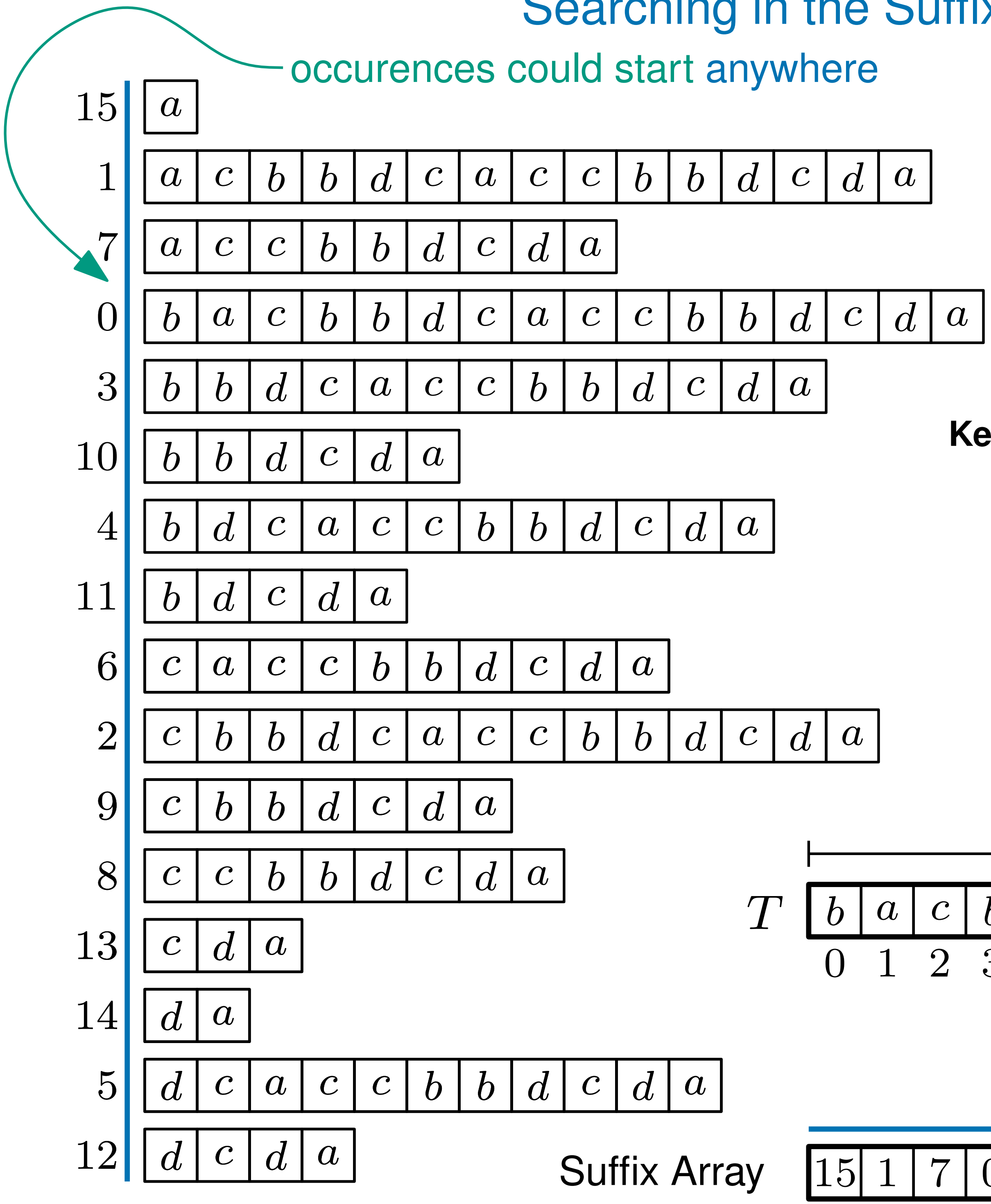
T (length n):

b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences could start anywhere

15	a
1	a c b b d c a c c b b d c d a
7	a c c b b d c d a
0	b a c b b d c a c c b b d c d a
3	b b d c a c c b b d c d a
10	b b d c d a
4	b d c a c c b b d c d a
11	b d c d a
6	c a c c b b d c d a
2	c b b d c a c c b b d c d a
9	c b b d c d a
8	c c b b d c d a
13	c d a
14	d a
5	d c a c c b b d c d a
12	d c d a

find P (length m)

c	b	b	d	c

Key Idea:

Find an occurence of P

using binary search

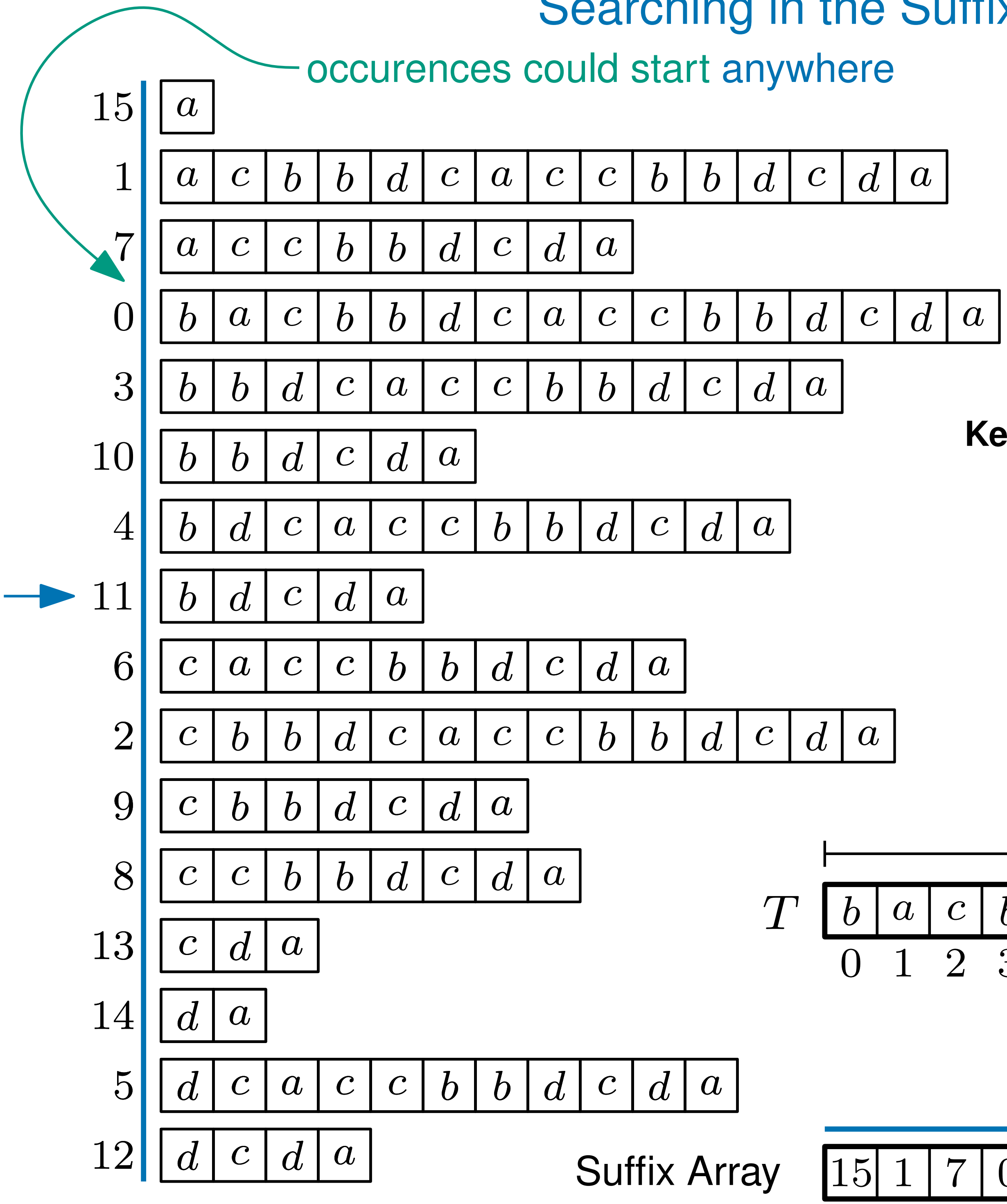
T (length n)

b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences could start anywhere

15	a
1	$a\,c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
7	$a\,c\,c\,b\,b\,d\,c\,d\,a$
0	$b\,a\,c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
3	$b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
10	$b\,b\,d\,c\,d\,a$
4	$b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
11	$b\,d\,c\,d\,a$
6	$c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
2	$c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
9	$c\,b\,b\,d\,c\,d\,a$
8	$c\,c\,b\,b\,d\,c\,d\,a$
13	$c\,d\,a$
14	$d\,a$
5	$d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
12	$d\,c\,d\,a$

find P

$$\overbrace{\boxed{c\,|\,b\,|\,b\,|\,d\,|\,c}}^{m}$$

Key Idea:

Find an occurence of P

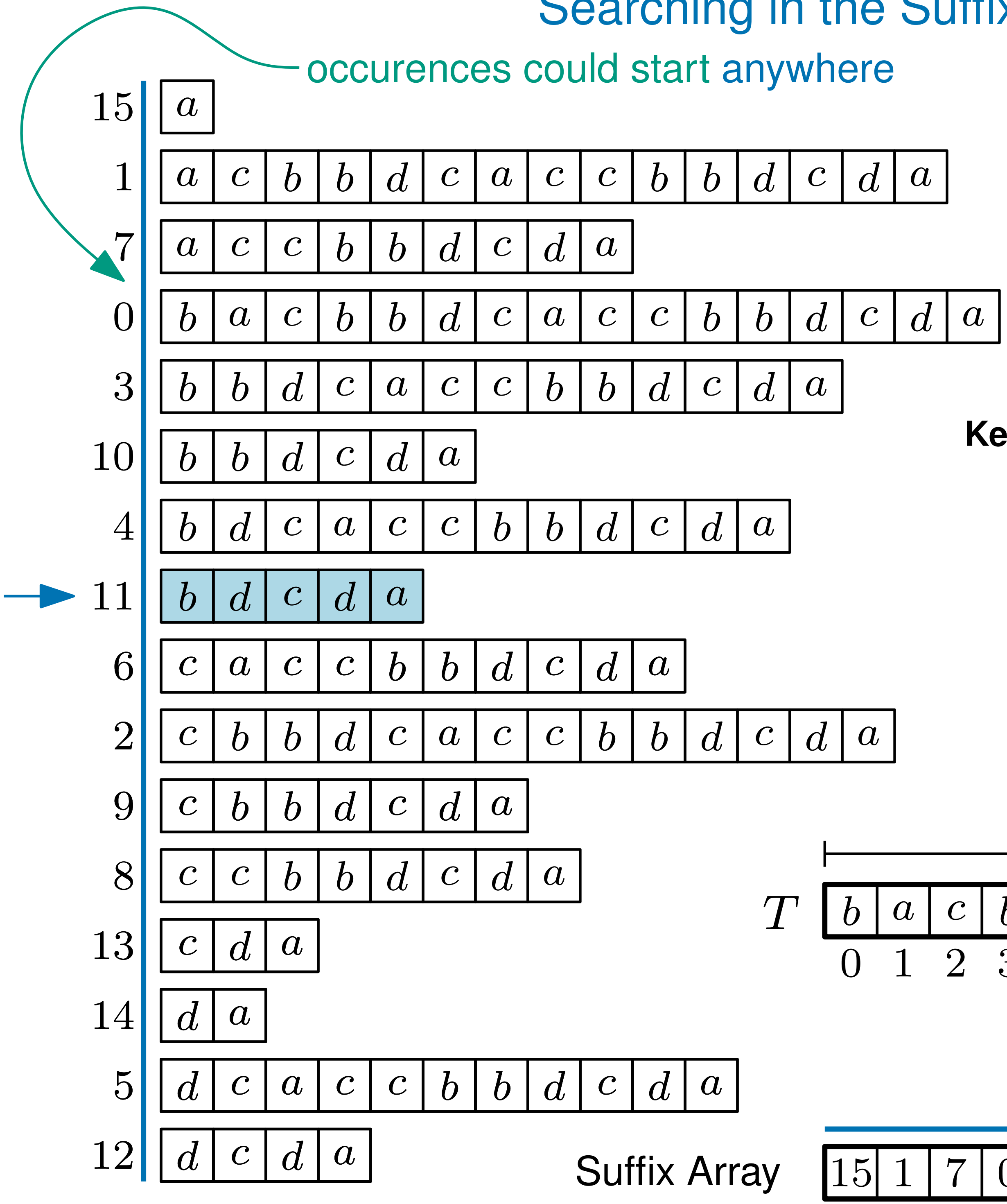
using binary search

$$\overbrace{\hphantom{b\,a\,c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a}}^{n}$$

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

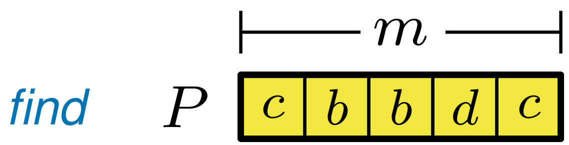
Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences could start anywhere

15	a
1	acbbdcaccbbdcda
7	accbbdcda
0	bacbbdcaccbbdcda
3	bbdcaccbbdcda
10	bbdcda
4	bdcaccbbdcda
11	bdcda
6	caccbbdcda
2	cbbdcaccbbdcda
9	cbbdcda
8	ccbbdcda
13	cda
14	da
5	dcaccbbdcda
12	dcda

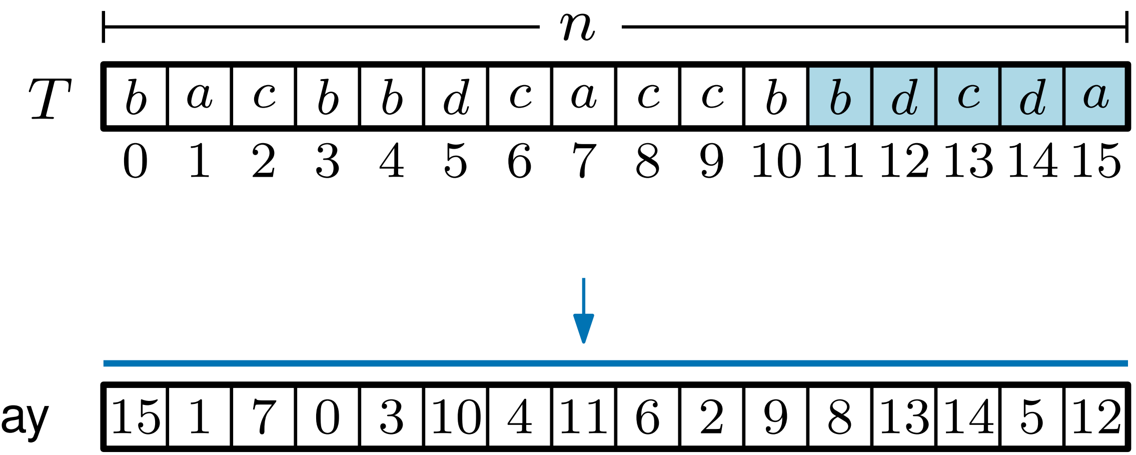


find P = c b b d c (m)

Key Idea:

Find an occurence of P

using binary search



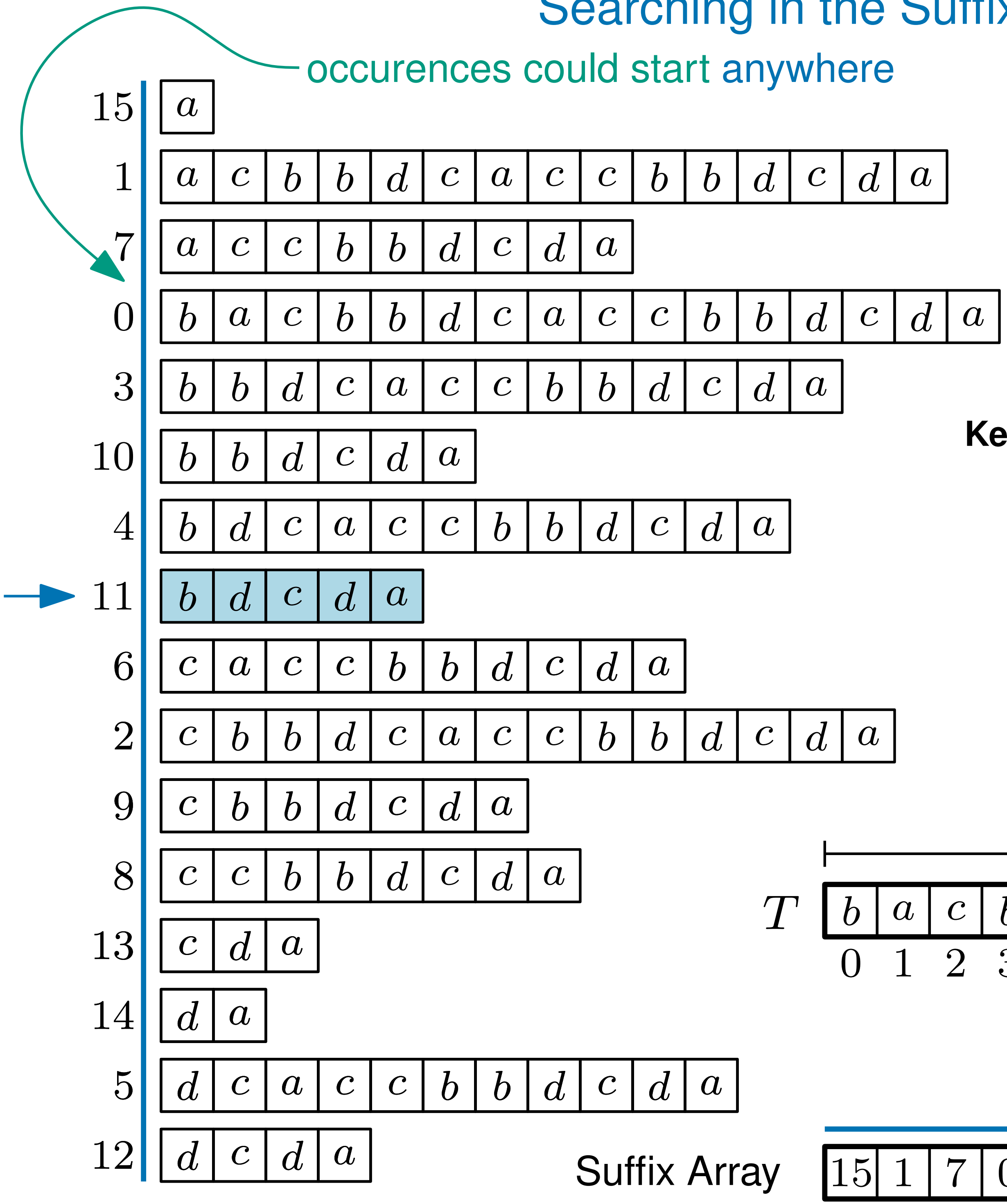
T (n)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences could start anywhere

15 a
1 $a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
7 $a\ c\ c\ b\ b\ d\ c\ d\ a$
0 $b\ a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
3 $b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
10 $b\ b\ d\ c\ d\ a$
4 $b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
11 $b\ d\ c\ d\ a$
6 $c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
2 $c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
9 $c\ b\ b\ d\ c\ d\ a$
8 $c\ c\ b\ b\ d\ c\ d\ a$
13 $c\ d\ a$
14 $d\ a$
5 $d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
12 $d\ c\ d\ a$

find P $\overbrace{c\ b\ b\ d\ c}^{m}$

Key Idea:

Find an occurence of P

using binary search

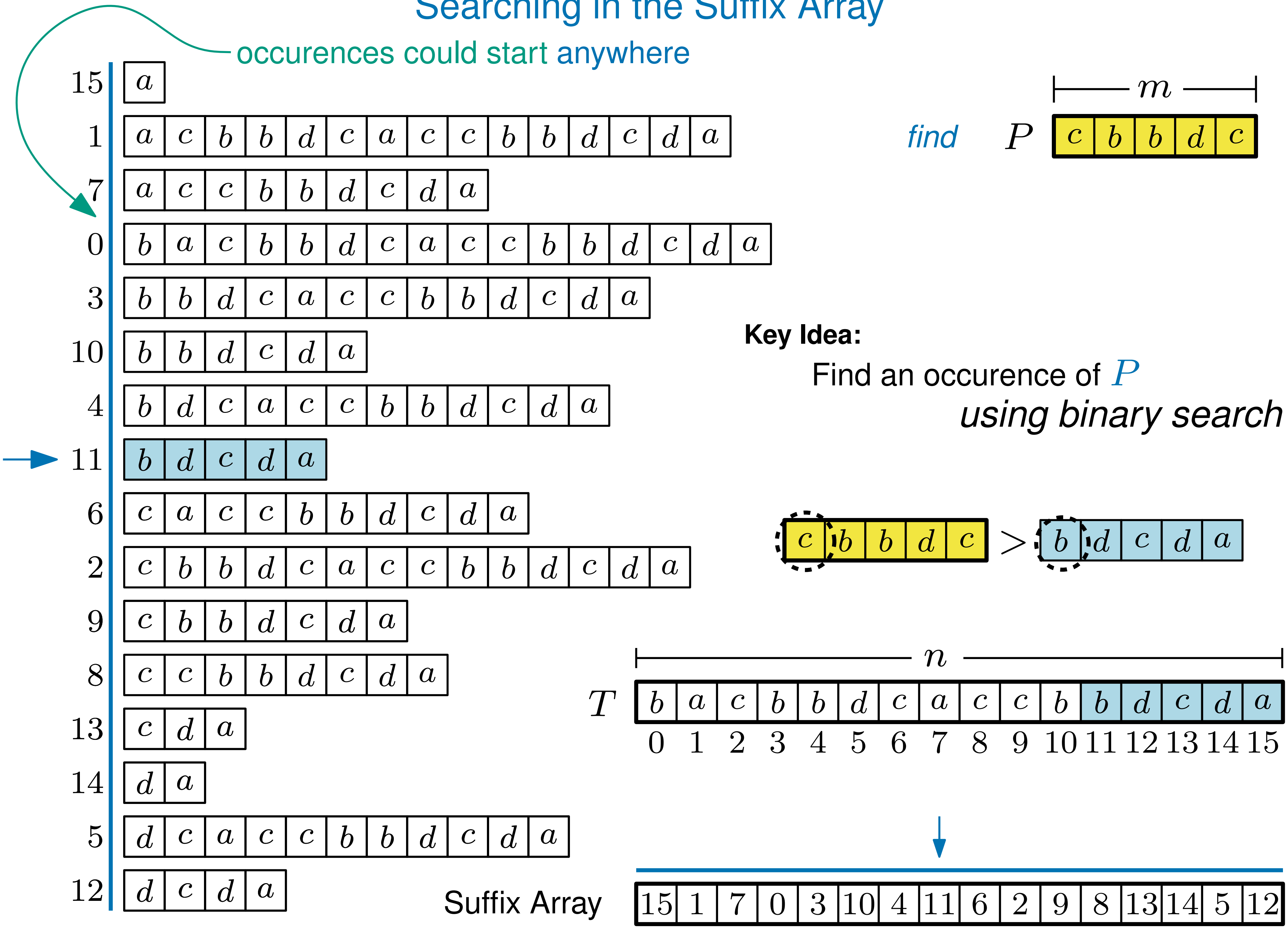
$c\ b\ b\ d\ c > b\ d\ c\ d\ a$

n

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



15	a
1	$a\,c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
7	$a\,c\,c\,b\,b\,d\,c\,d\,a$
0	$b\,a\,c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
3	$b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
10	$b\,b\,d\,c\,d\,a$
4	$b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
11	$b\,d\,c\,d\,a$
6	$c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
2	$c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
9	$c\,b\,b\,d\,c\,d\,a$
8	$c\,c\,b\,b\,d\,c\,d\,a$
13	$c\,d\,a$
14	$d\,a$
5	$d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
12	$d\,c\,d\,a$

find P = $c\,b\,b\,d\,c$

Key Idea:

Find an occurence of P

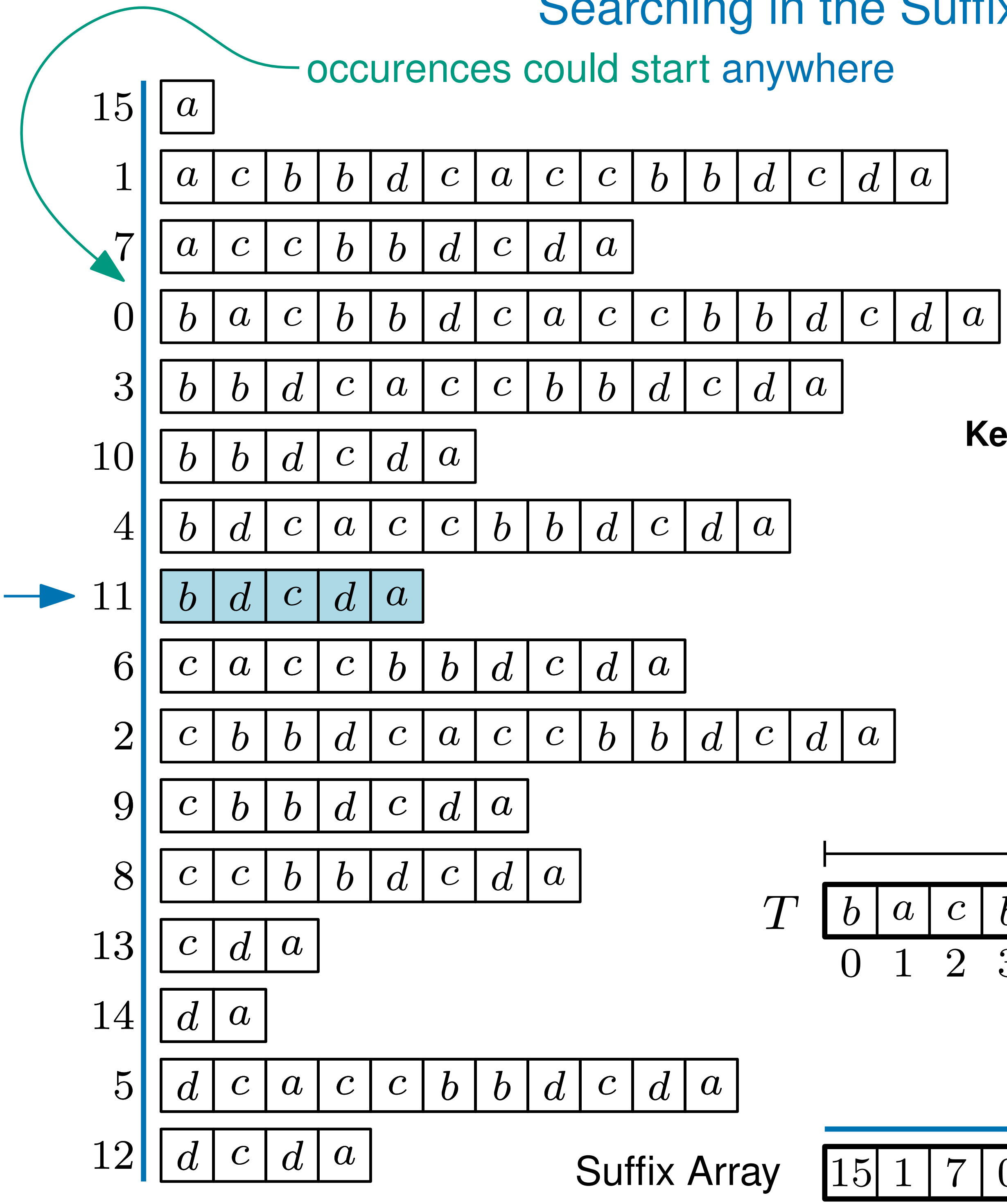
using binary search

$c\,b\,b\,d\,c > b\,d\,c\,d\,a$

T																
	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array																
	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



15	a
1	$acbbdcaccbbdcda$
7	$accbbdcda$
0	$bacbbdcaccbbdcda$
3	$bbdcaccbbdcda$
10	$bbdcda$
4	$bdcaccbbdcda$
11	$bdcda$
6	$caccbbdcda$
2	$cbbdcaccbbdcda$
9	$cbbdcda$
8	$ccbbdcda$
13	cda
14	da
5	$dcaccbbdcda$
12	$dcda$

find P = $cbbdc$ (m)

Key Idea:

Find an occurence of P

using binary search

$cbbdc > bdcda$

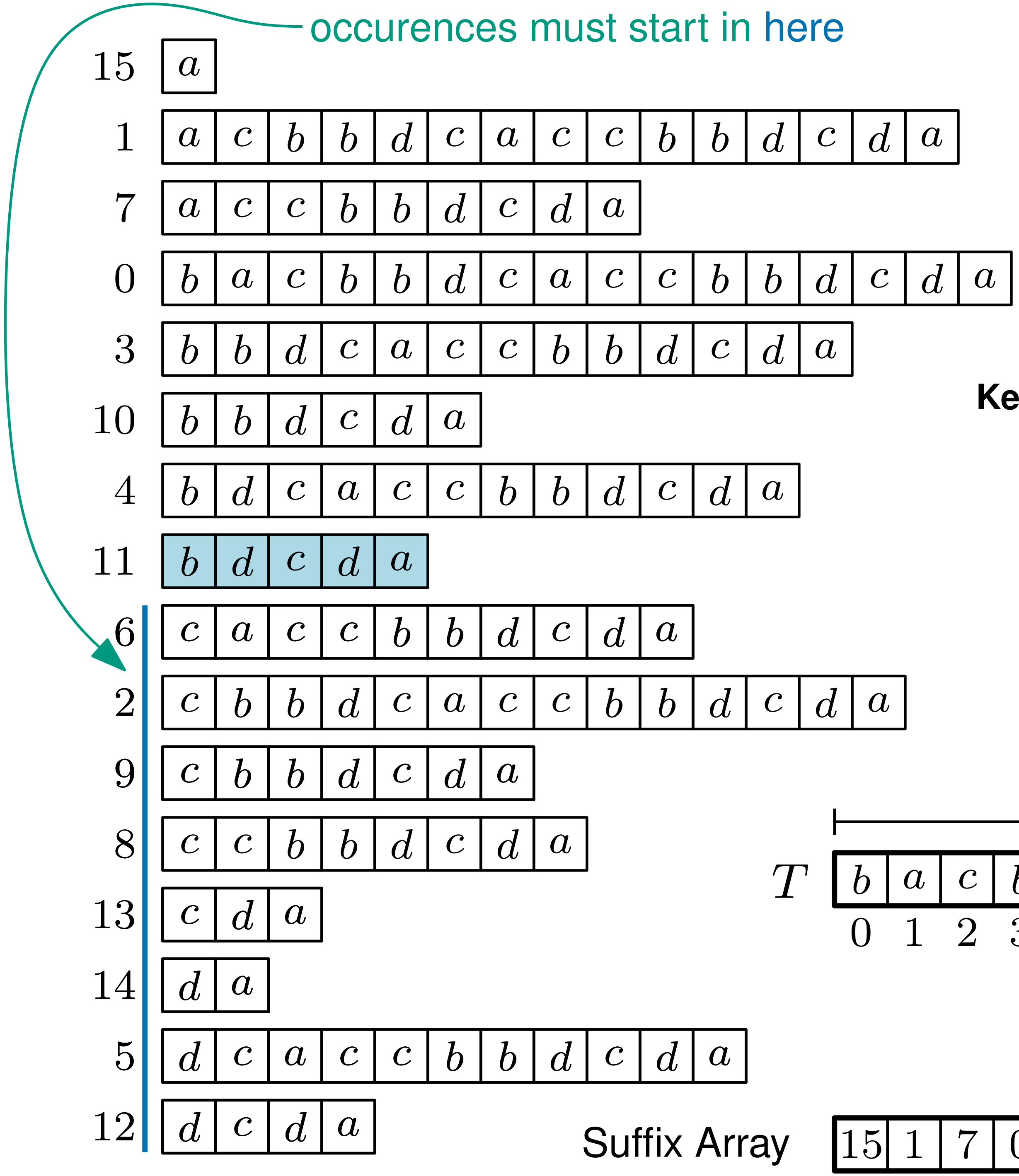
T (n)

b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences must start in here

15	a															
1	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a	
7	a	c	c	b	b	d	c	d	a							
0	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
3	b	b	d	c	a	c	c	b	b	d	c	d	a			
10	b	b	d	c	d	a										
4	b	d	c	a	c	c	b	b	d	c	d	a				
11	b	d	c	d	a											
6	c	a	c	c	b	b	d	c	d	a						
2	c	b	b	d	c	a	c	c	b	b	d	c	d	a		
9	c	b	b	d	c	d	a									
8	c	c	b	b	d	c	d	a								
13	c	d	a													
14	d	a														
5	d	c	a	c	c	b	b	d	c	d	a					
12	d	c	d	a												

find P = $c\ b\ b\ d\ c$ (length m)

Key Idea:

Find an occurence of P

using binary search

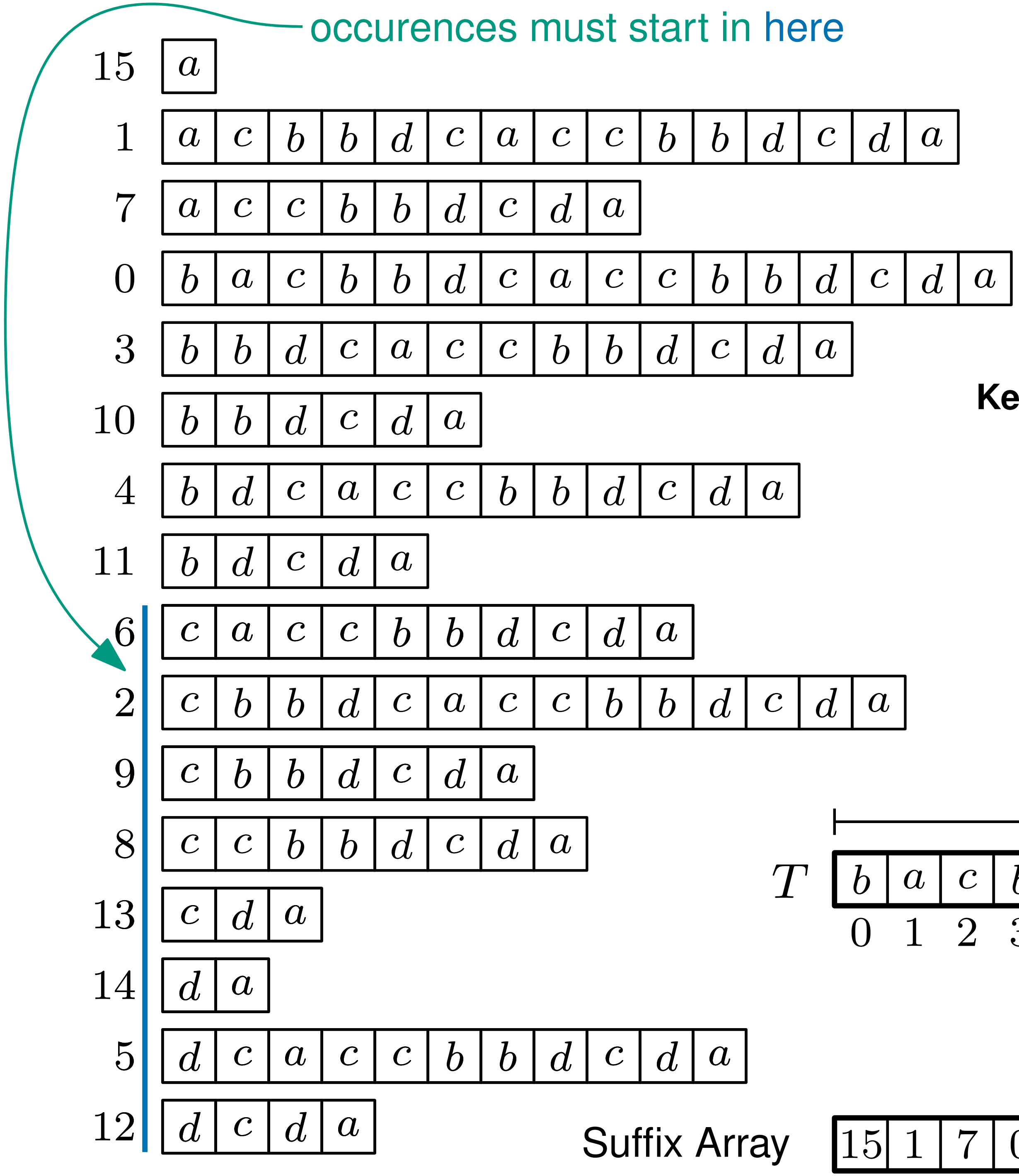
$c\ b\ b\ d\ c > b\ d\ c\ d\ a$

n

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences must start in here

15 a

1 $a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$

7 $a\ c\ c\ b\ b\ d\ c\ d\ a$

0 $b\ a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$

3 $b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$

10 $b\ b\ d\ c\ d\ a$

4 $b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$

11 $b\ d\ c\ d\ a$

6 $c\ a\ c\ c\ b\ b\ d\ c\ d\ a$

2 $c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$

9 $c\ b\ b\ d\ c\ d\ a$

8 $c\ c\ b\ b\ d\ c\ d\ a$

13 $c\ d\ a$

14 $d\ a$

5 $d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$

12 $d\ c\ d\ a$

find P $\overbrace{c\ b\ b\ d\ c}^{m}$

Key Idea:

Find an occurence of P

using binary search

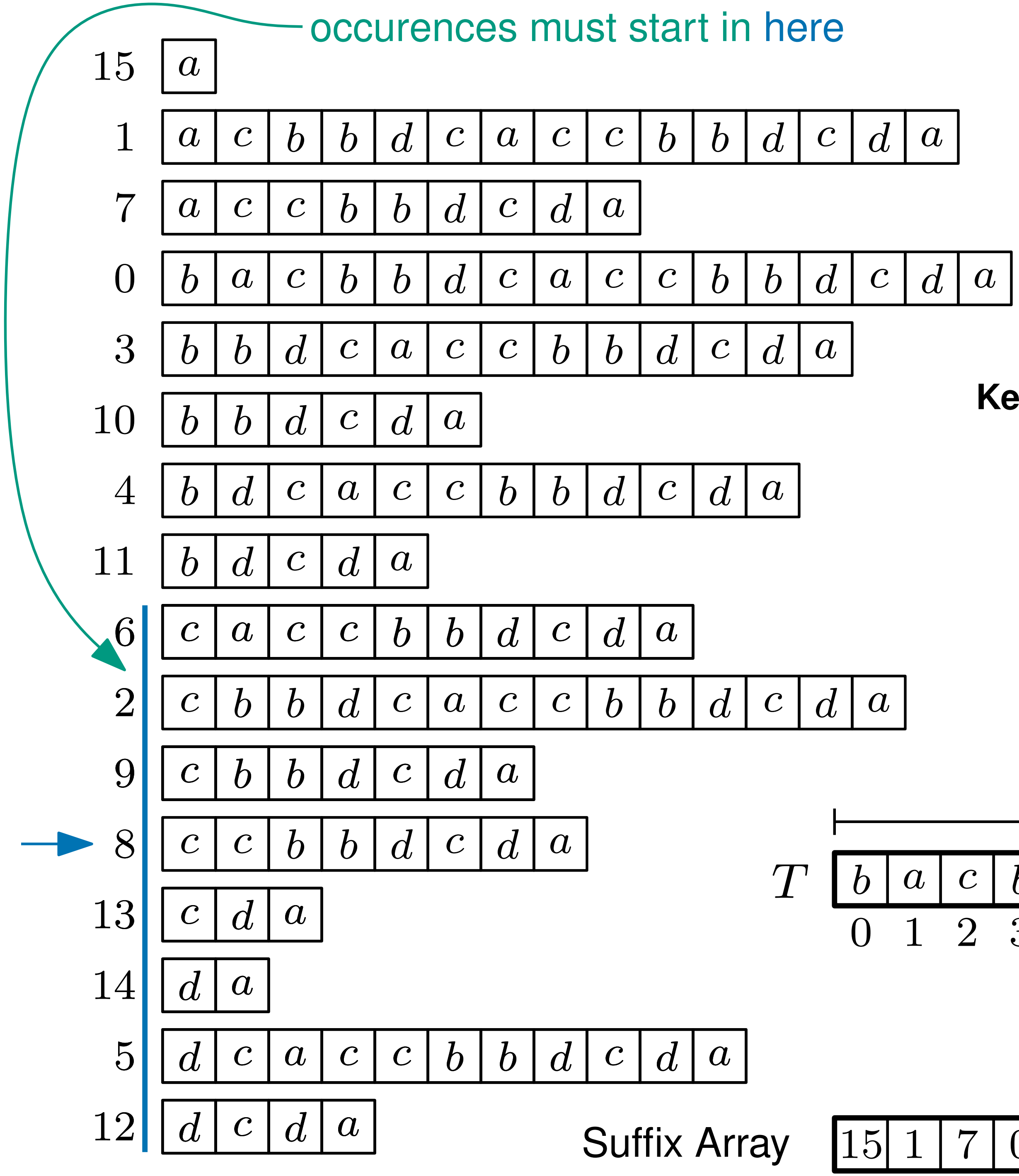
T, length n:

b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences must start in here

15	a
1	$a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
7	$a\ c\ c\ b\ b\ d\ c\ d\ a$
0	$b\ a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
3	$b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
10	$b\ b\ d\ c\ d\ a$
4	$b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
11	$b\ d\ c\ d\ a$
6	$c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
2	$c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
9	$c\ b\ b\ d\ c\ d\ a$
8	$c\ c\ b\ b\ d\ c\ d\ a$
13	$c\ d\ a$
14	$d\ a$
5	$d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
12	$d\ c\ d\ a$

find P $\overbrace{c\ b\ b\ d\ c}^{m}$

Key Idea:

Find an occurence of P

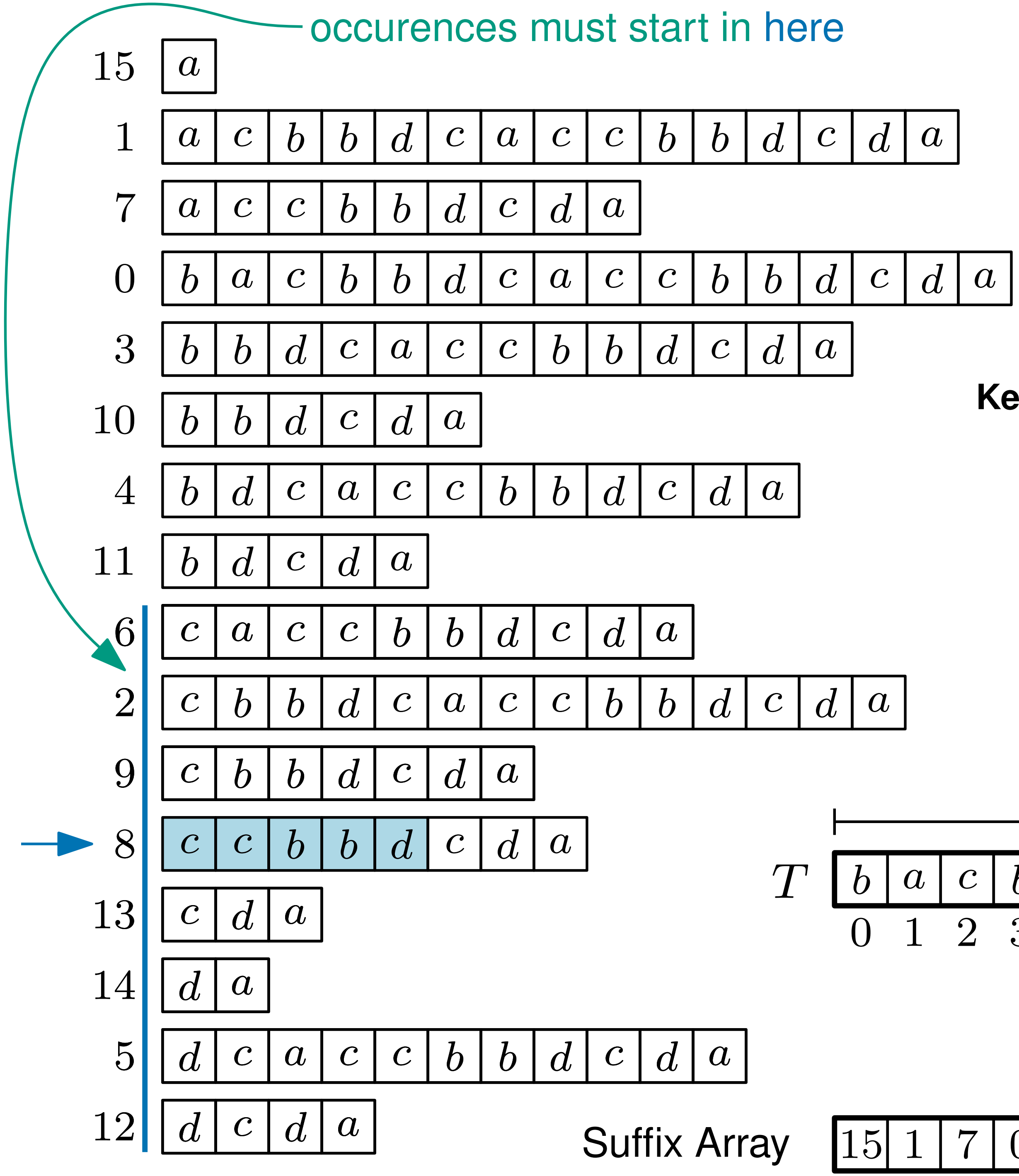
using binary search

n

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



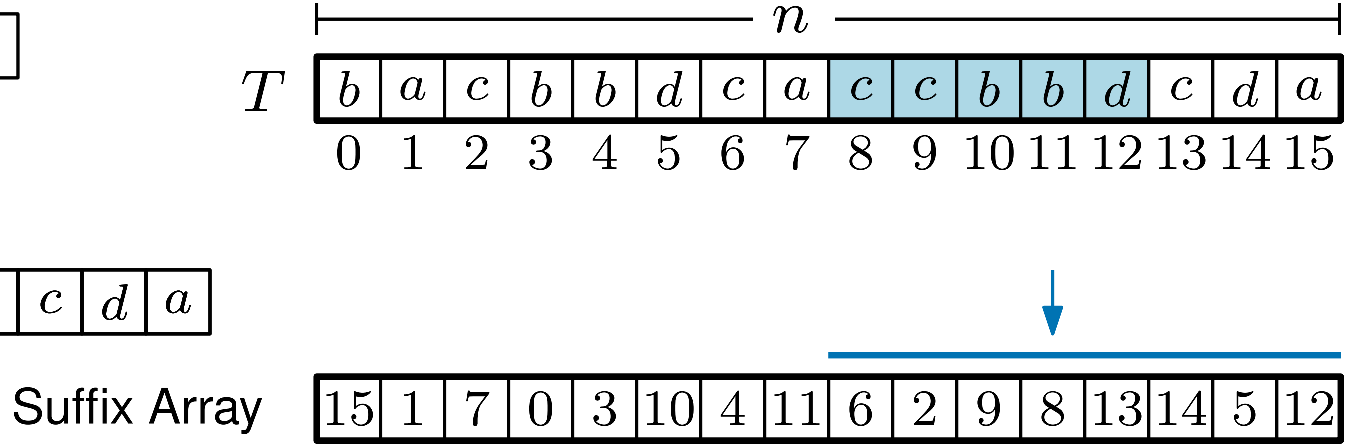
15	a
1	a c b b d c a c c b b d c d a
7	a c c b b d c d a
0	b a c b b d c a c c b b d c d a
3	b b d c a c c b b d c d a
10	b b d c d a
4	b d c a c c b b d c d a
11	b d c d a
6	c a c c b b d c d a
2	c b b d c a c c b b d c d a
9	c b b d c d a
8	c c b b d c d a
13	c d a
14	d a
5	d c a c c b b d c d a
12	d c d a

find P = c b b d c (length m)

Key Idea:

Find an occurence of P

using binary search

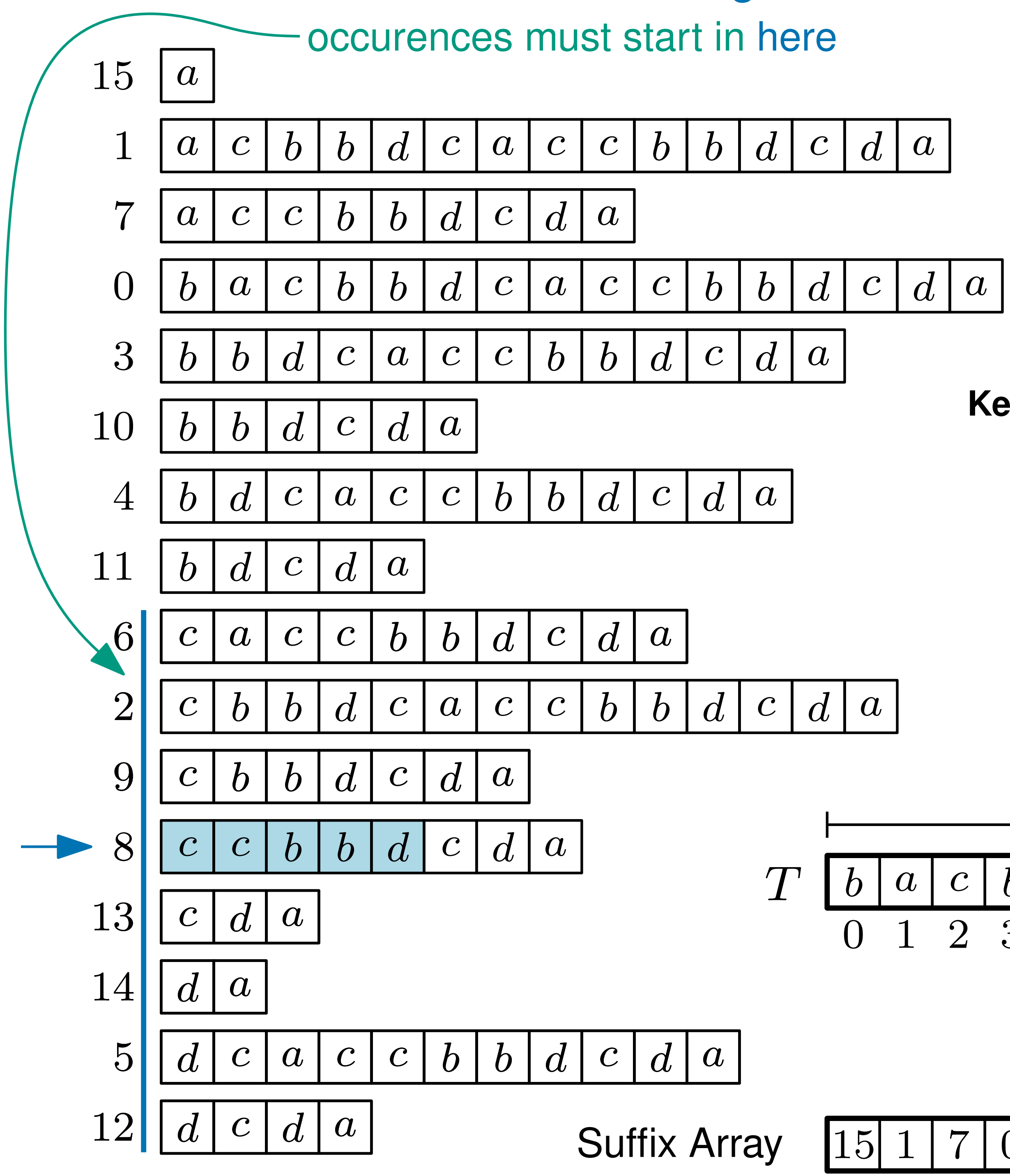


T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

n

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences must start in here

15	a
1	$a\,c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
7	$a\,c\,c\,b\,b\,d\,c\,d\,a$
0	$b\,a\,c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
3	$b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
10	$b\,b\,d\,c\,d\,a$
4	$b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
11	$b\,d\,c\,d\,a$
6	$c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
2	$c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
9	$c\,b\,b\,d\,c\,d\,a$
8	$c\,c\,b\,b\,d\,c\,d\,a$
13	$c\,d\,a$
14	$d\,a$
5	$d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
12	$d\,c\,d\,a$

find P $\overbrace{c\,b\,b\,d\,c}^{m}$

Key Idea:

Find an occurence of P

using binary search

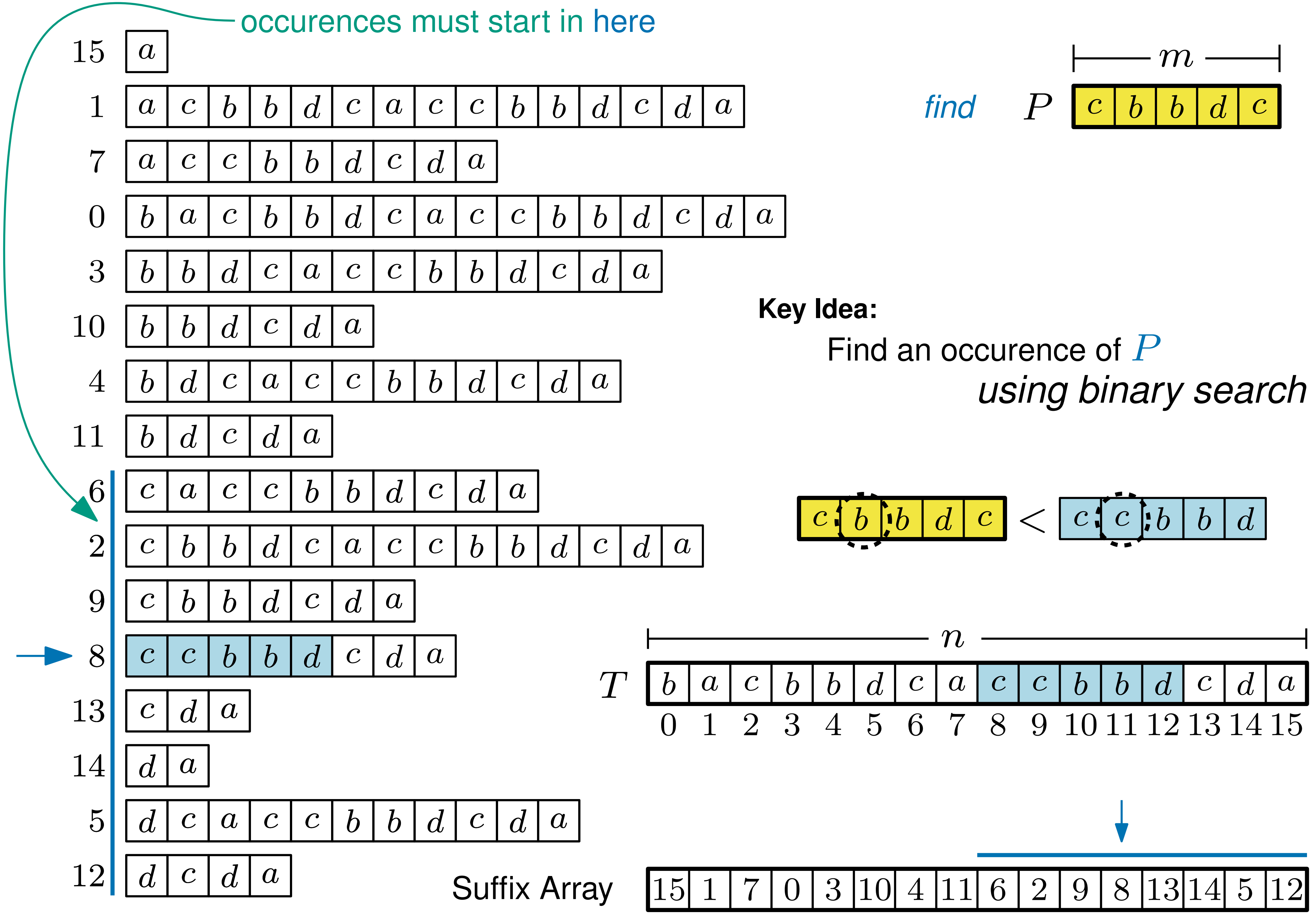
$c\,b\,b\,d\,c < c\,c\,b\,b\,d$

n

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences must start in here

15	a
1	$a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
7	$a\ c\ c\ b\ b\ d\ c\ d\ a$
0	$b\ a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
3	$b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
10	$b\ b\ d\ c\ d\ a$
4	$b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
11	$b\ d\ c\ d\ a$
6	$c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
2	$c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
9	$c\ b\ b\ d\ c\ d\ a$
8	$c\ c\ b\ b\ d\ c\ d\ a$
13	$c\ d\ a$
14	$d\ a$
5	$d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
12	$d\ c\ d\ a$

find P $\overbrace{c\ b\ b\ d\ c}^{m}$

Key Idea:

Find an occurence of P

using binary search

$c\ b\ b\ d\ c < c\ c\ b\ b\ d$

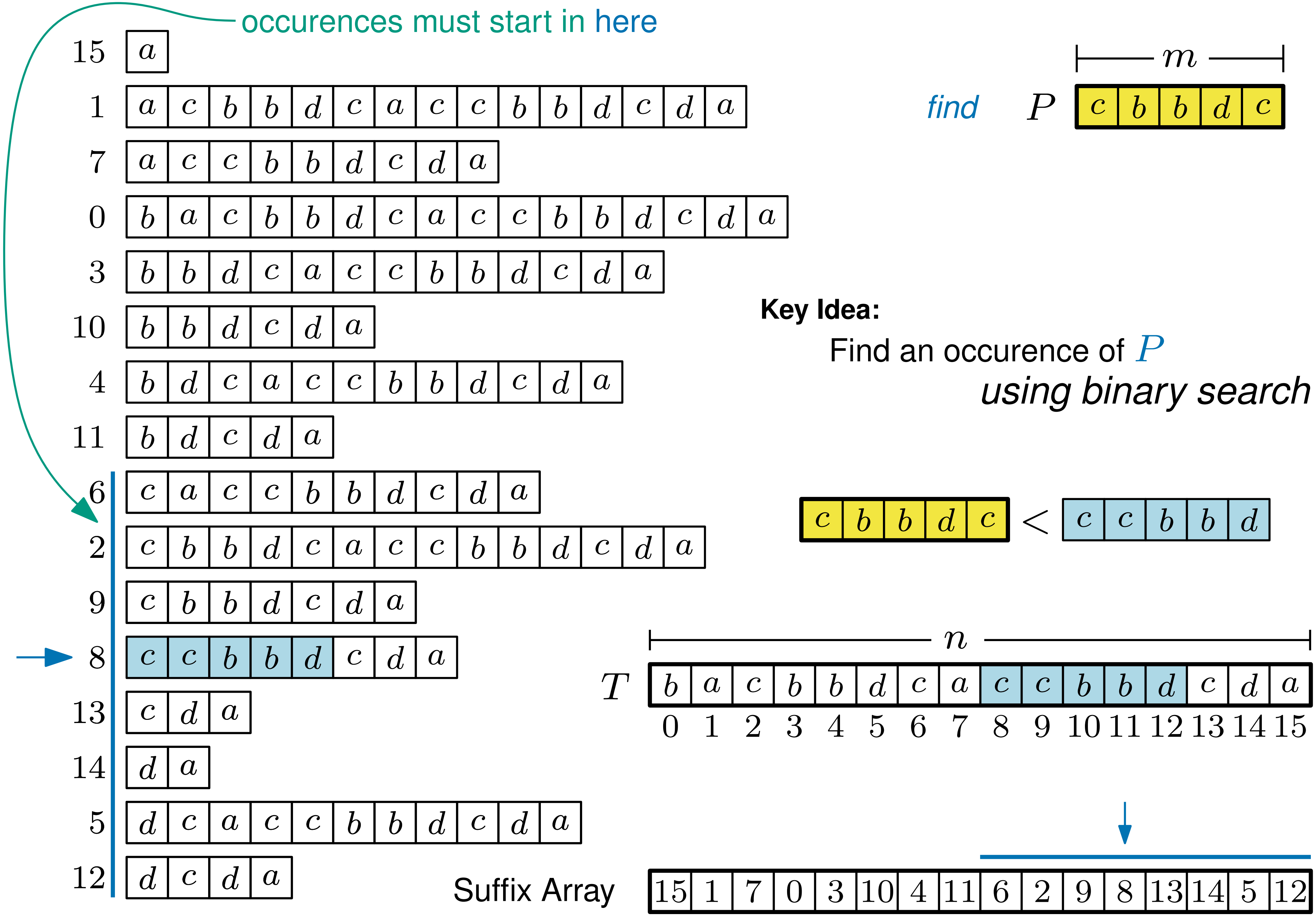
T (n)

b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences must start in here

15	a
1	$a\,c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
7	$a\,c\,c\,b\,b\,d\,c\,d\,a$
0	$b\,a\,c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
3	$b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
10	$b\,b\,d\,c\,d\,a$
4	$b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
11	$b\,d\,c\,d\,a$
6	$c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
2	$c\,b\,b\,d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
9	$c\,b\,b\,d\,c\,d\,a$
8	$c\,c\,b\,b\,d\,c\,d\,a$
13	$c\,d\,a$
14	$d\,a$
5	$d\,c\,a\,c\,c\,b\,b\,d\,c\,d\,a$
12	$d\,c\,d\,a$

find P = $c\,b\,b\,d\,c$ (m)

Key Idea:

Find an occurence of P

using binary search

$c\,b\,b\,d\,c < c\,c\,b\,b\,d$

n

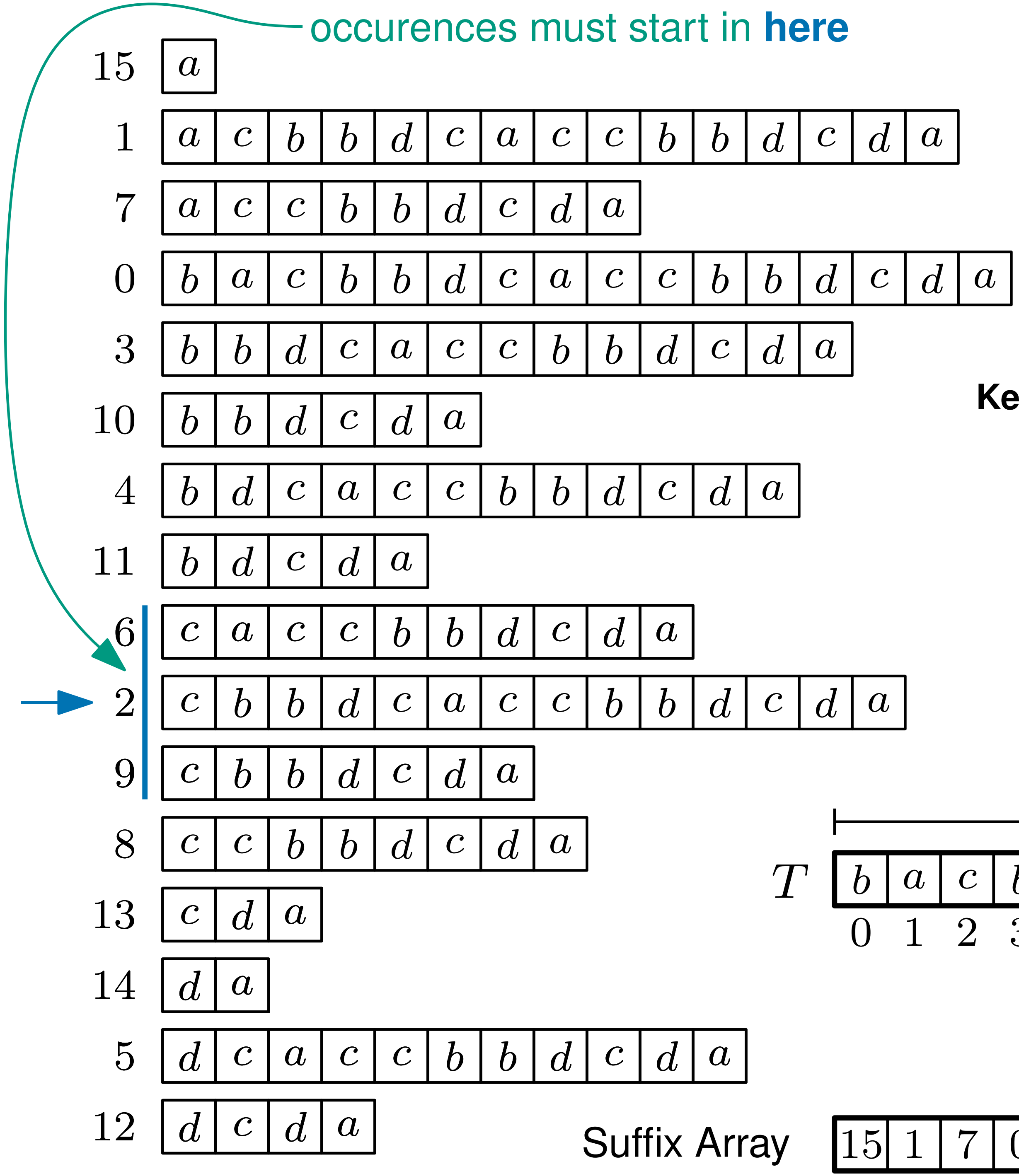
T

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



15	a
1	$a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
7	$a\ c\ c\ b\ b\ d\ c\ d\ a$
0	$b\ a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
3	$b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
10	$b\ b\ d\ c\ d\ a$
4	$b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
11	$b\ d\ c\ d\ a$
6	$c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
2	$c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
9	$c\ b\ b\ d\ c\ d\ a$
8	$c\ c\ b\ b\ d\ c\ d\ a$
13	$c\ d\ a$
14	$d\ a$
5	$d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
12	$d\ c\ d\ a$

find P = $c\ b\ b\ d\ c$ (m)

Key Idea:

Find an occurence of P

using binary search

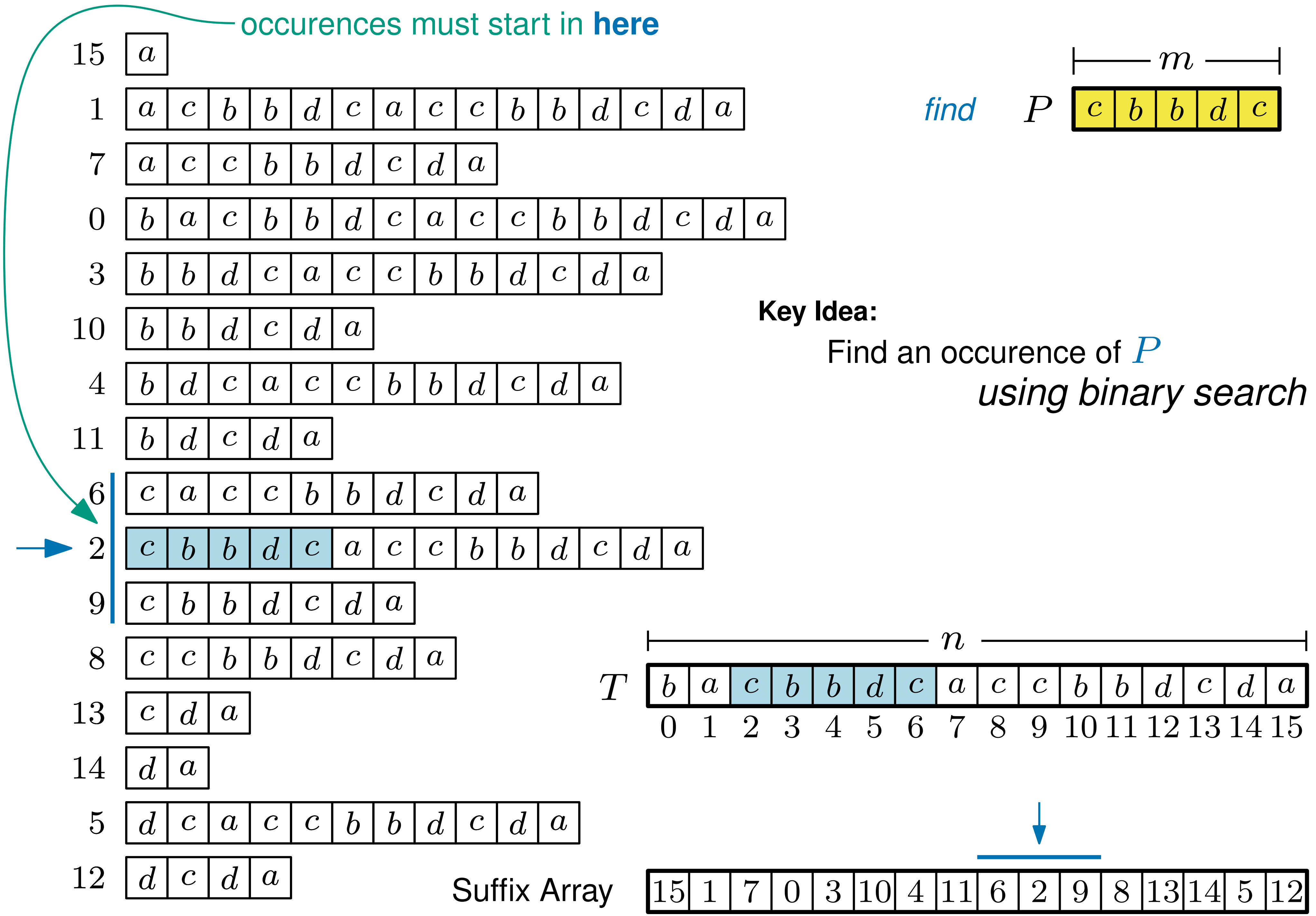
T (n)

b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences must start in **here**

15	a
1	$acbbdcaccbbdcda$
7	$accbbdcda$
0	$bacbbdcaccbbdcda$
3	$bbdcaccbbdcda$
10	$bbdcda$
4	$bdcaccbbdcda$
11	$bdcda$
6	$caccbbdcda$
2	$cbbdcaccbbdcda$
9	$cbbdcda$
8	$ccbbdcda$
13	cda
14	da
5	$dcaccbbdcda$
12	$dcda$

find P = $cbbdc$ (m)

Key Idea:

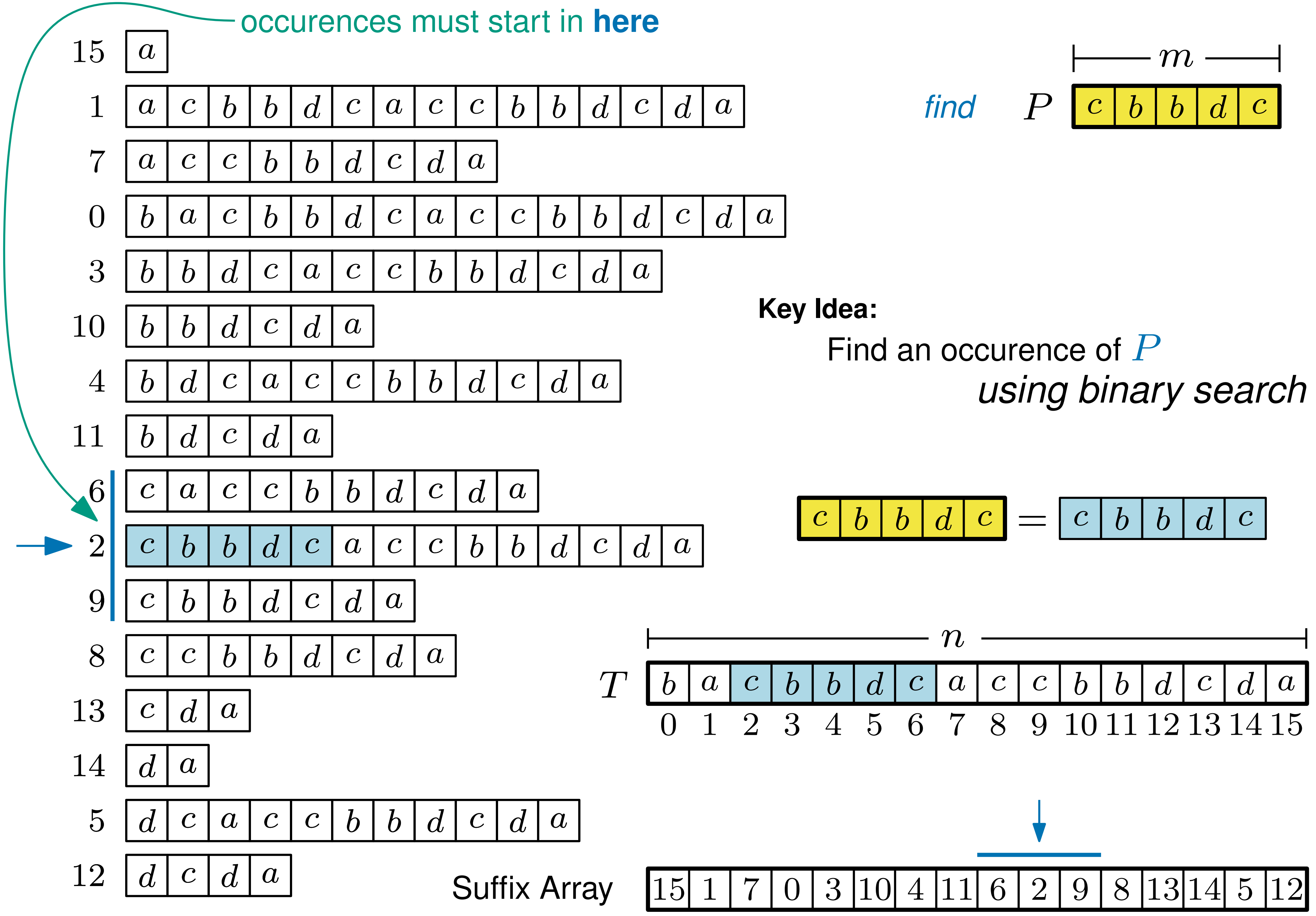
Find an occurence of P *using binary search*

n

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences must start in **here**

15	a
1	a c b b d c a c c b b d c d a
7	a c c b b d c d a
0	b a c b b d c a c c b b d c d a
3	b b d c a c c b b d c d a
10	b b d c d a
4	b d c a c c b b d c d a
11	b d c d a
6	c a c c b b d c d a
2	c b b d c a c c b b d c d a
9	c b b d c d a
8	c c b b d c d a
13	c d a
14	d a
5	d c a c c b b d c d a
12	d c d a

find P c b b d c (m)

Key Idea:

Find an occurence of P

using binary search

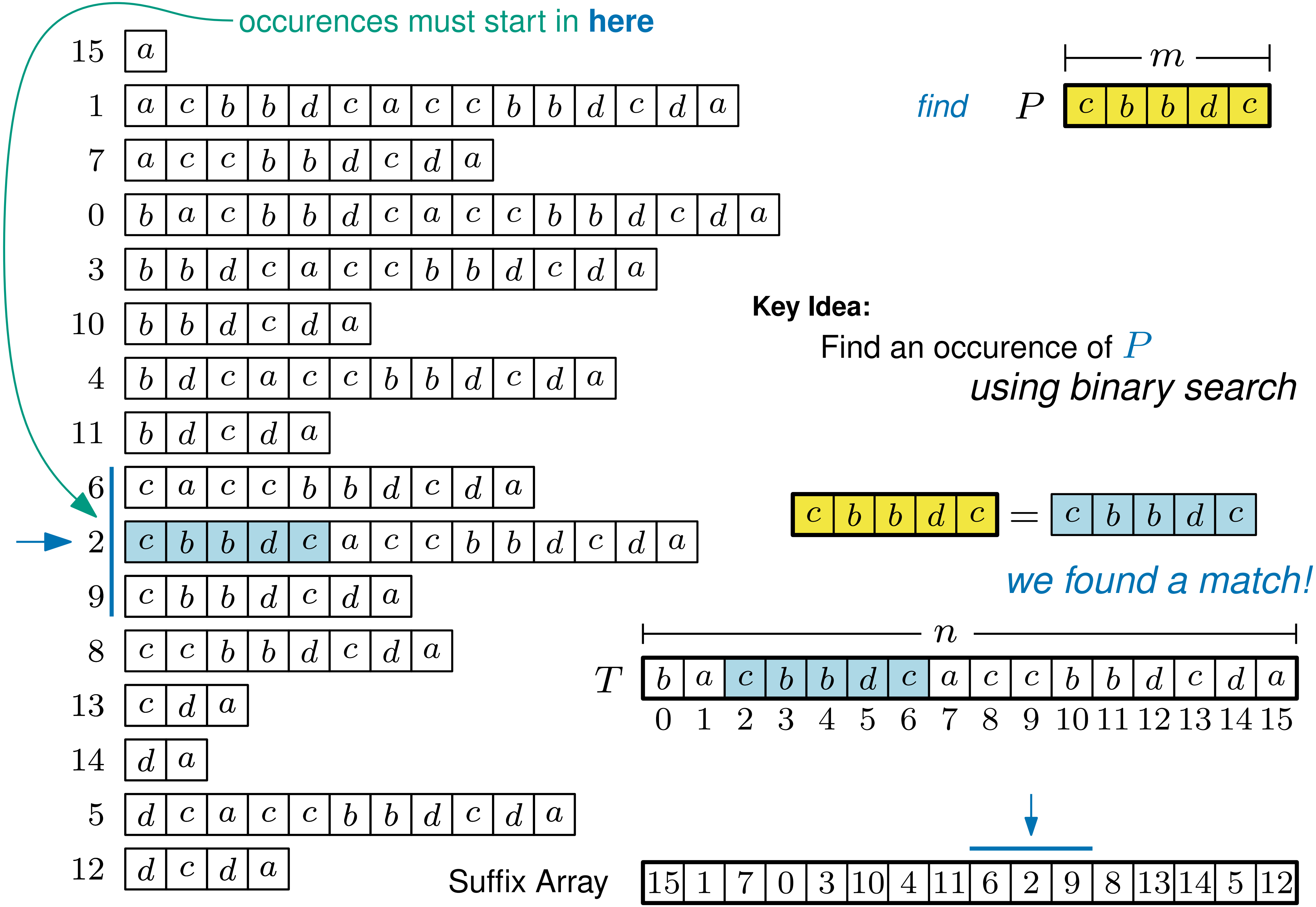
c b b d c = c b b d c

n

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



occurences must start in **here**

15	a
1	$acbbdcaccbbdcda$
7	$accbbdcda$
0	$bacbbdcaccbbdcda$
3	$bbdcaccbbdcda$
10	$bbdcda$
4	$bdcaccbbdcda$
11	$bdcda$
6	$caccbbdcda$
2	$cbbdcaccbbdcda$
9	$cbbdcda$
8	$ccbbdcda$
13	cda
14	da
5	$dcaccbbdcda$
12	$dcda$

find P = $cbbdc$ (m)

Key Idea:

Find an occurence of P

using binary search

$cbbdc = cbbdc$

we found a match!

T (n)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array

15	a															
1	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a	
7	a	c	c	b	b	d	c	d	a							
0	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
3	b	b	d	c	a	c	c	b	b	d	c	d	a			
10	b	b	d	c	d	a										
4	b	d	c	a	c	c	b	b	d	c	d	a				
11	b	d	c	d	a											
6	c	a	c	c	b	b	d	c	d	a						
2	c	b	b	d	c	a	c	c	b	b	d	c	d	a		
9	c	b	b	d	c	d	a									
8	c	c	b	b	d	c	d	a								
13	c	d	a													
14	d	a														
5	d	c	a	c	c	b	b	d	c	d	a					
12	d	c	d	a												

m

find P

c	b	b	d	c

Key Idea:

Find an occurence of P

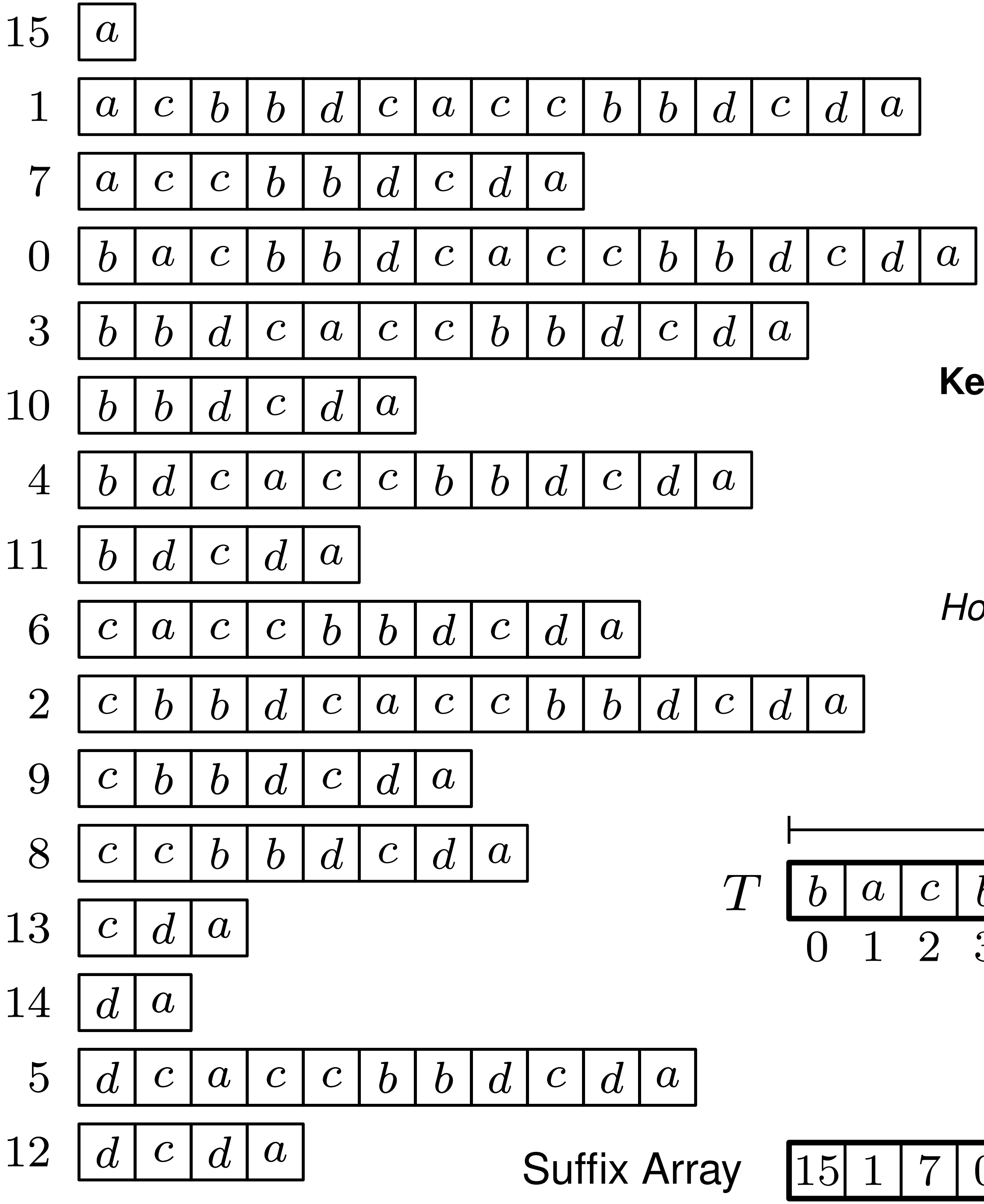
using binary search

n

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



15	a
1	$a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
7	$a\ c\ c\ b\ b\ d\ c\ d\ a$
0	$b\ a\ c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
3	$b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
10	$b\ b\ d\ c\ d\ a$
4	$b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
11	$b\ d\ c\ d\ a$
6	$c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
2	$c\ b\ b\ d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
9	$c\ b\ b\ d\ c\ d\ a$
8	$c\ c\ b\ b\ d\ c\ d\ a$
13	$c\ d\ a$
14	$d\ a$
5	$d\ c\ a\ c\ c\ b\ b\ d\ c\ d\ a$
12	$d\ c\ d\ a$

m

find P $\boxed{c\ b\ b\ d\ c}$

Key Idea:

Find an occurence of P

using binary search

How long does this take?

n

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array

15	a															
1	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a	
7	a	c	c	b	b	d	c	d	a							
0	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
3	b	b	d	c	a	c	c	b	b	d	c	d	a			
10	b	b	d	c	d	a										
4	b	d	c	a	c	c	b	b	d	c	d	a				
11	b	d	c	d	a											
6	c	a	c	c	b	b	d	c	d	a						
2	c	b	b	d	c	a	c	c	b	b	d	c	d	a		
9	c	b	b	d	c	d	a									
8	c	c	b	b	d	c	d	a								
13	c	d	a													
14	d	a														
5	d	c	a	c	c	b	b	d	c	d	a					
12	d	c	d	a												

m

find P

c	b	b	d	c

Key Idea:

Find an occurence of P

using binary search

How long does this take?

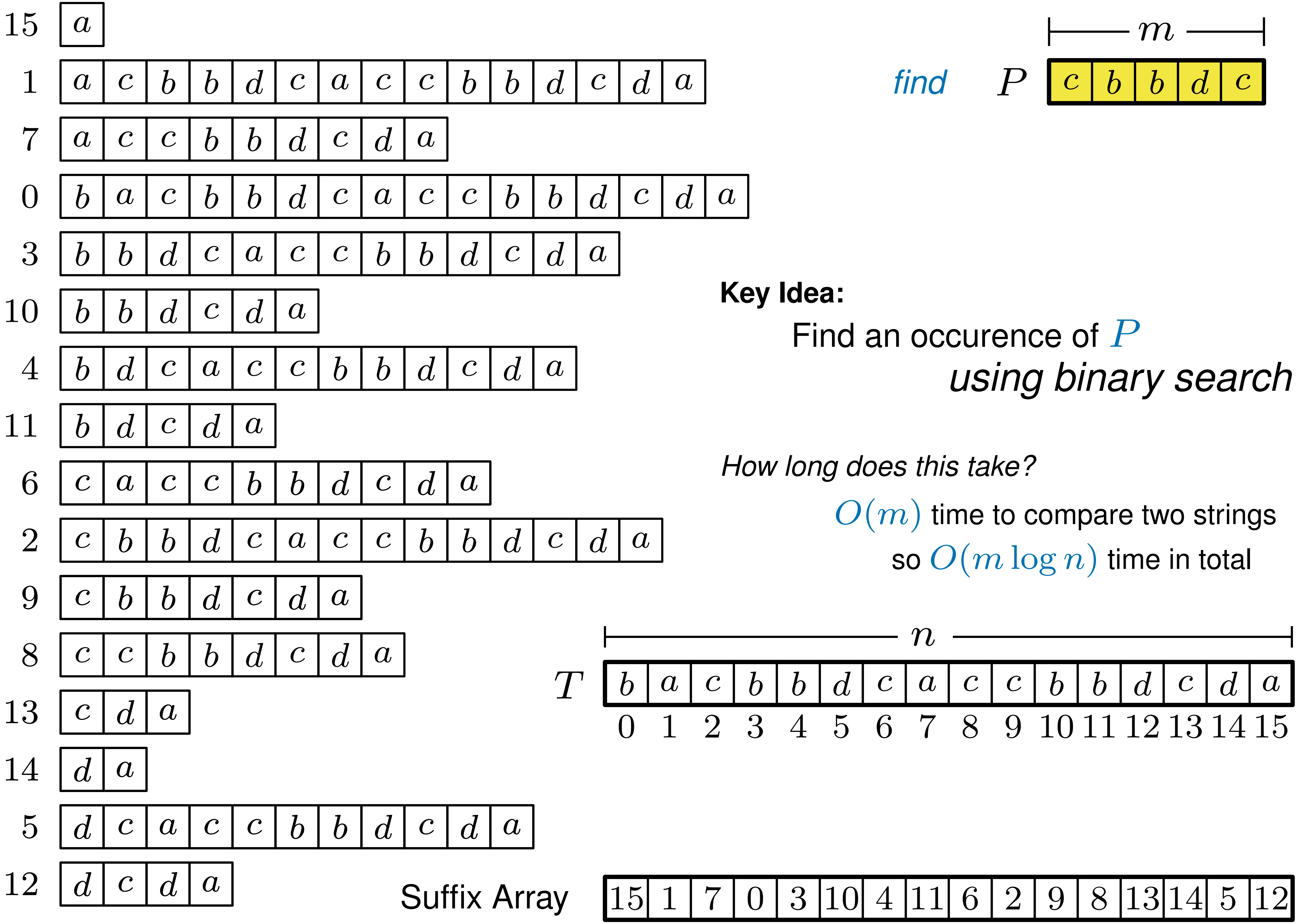
$O(m)$ time to compare two strings

n

T	b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array	15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array



find P = c b b d c (m)

15	a
1	a c b b d c a c c b b d c d a
7	a c c b b d c d a
0	b a c b b d c a c c b b d c d a
3	b b d c a c c b b d c d a
10	b b d c d a
4	b d c a c c b b d c d a
11	b d c d a
6	c a c c b b d c d a
2	c b b d c a c c b b d c d a
9	c b b d c d a
8	c c b b d c d a
13	c d a
14	d a
5	d c a c c b b d c d a
12	d c d a

Key Idea:

Find an occurence of P

using binary search

How long does this take?

$O(m)$ time to compare two strings

so $O(m \log n)$ time in total

T (n)

b	a	c	b	b	d	c	a	c	c	b	b	d	c	d	a
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

Searching in the Suffix Array

15	a
1	a c b b d c a c c b b d c d a
7	a c c b b d c d a
0	b a c b b d c a c c b b d c d a
3	b b d c a c c b b d c d a
10	b b d c d a
4	b d c a c c b b d c d a
11	b d c d a
6	c a c c b b d c d a
2	c b b d c a c c b b d c d a
9	c b b d c d a
8	c c b b d c d a
13	c d a
14	d a
5	d c a c c b b d c d a
12	d c d a

m

find P c b b d c

Key Idea:

Find an occurence of P

using binary search

How long does this take?

$O(m)$ time to compare two strings

so $O(m \log n)$ time in total

This method generalises to $O(m \log n + \text{occ})$ time

to find all occ occurences.

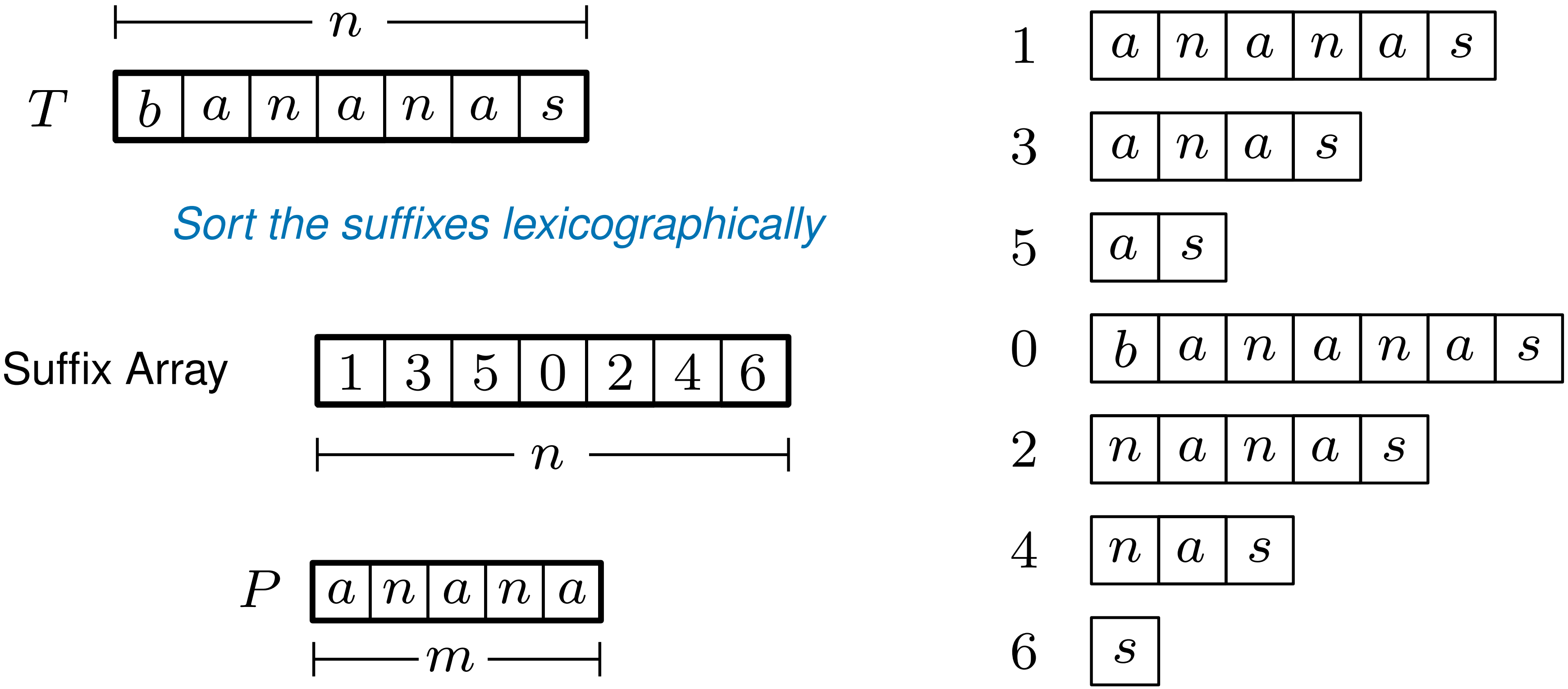
by continuing the binary search

(we will skip the details)

Suffix Array

15	1	7	0	3	10	4	11	6	2	9	8	13	14	5	12

The suffix array

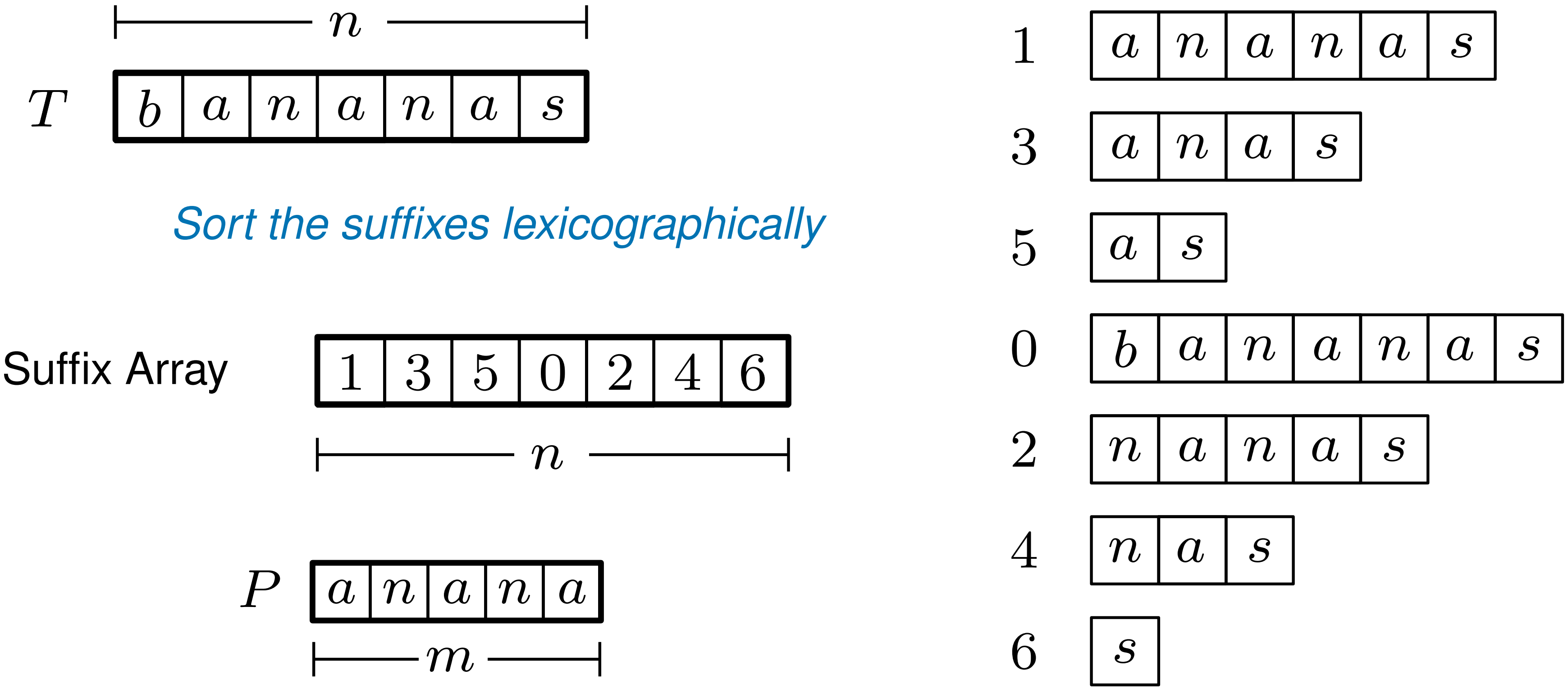


Finding an occurrence of a pattern (length m) takes $O(m \log n)$ time

Finding all occurrences takes $O(m \log n + \text{occ})$ time

where occ is the number of occurences

The suffix array



n

T | b | a | n | a | n | a | s

Sort the suffixes lexicographically

Suffix Array | 1 | 3 | 5 | 0 | 2 | 4 | 6

n

P | a | n | a | n | a

m

1	$a\ n\ a\ n\ a\ s$
3	$a\ n\ a\ s$
5	$a\ s$
0	$b\ a\ n\ a\ n\ a\ s$
2	$n\ a\ n\ a\ s$
4	$n\ a\ s$
6	s

Finding an occurrence of a pattern (length m) takes $O(m \log n)$ time

Finding all occurrences takes $O(m \log n + \text{occ})$ time

where occ is the number of occurences

This can be further improved to $O(m + \log n + \text{occ})$ time

(using LCP queries which we will see in a future lecture)

The suffix array

n

T	b	a	n	a	n	a	s

Sort the suffixes lexicographically

Suffix Array	1	3	5	0	2	4	6

n

P	a	n	a	n	a

m

1	a	n	a	n	a	s	
3	a	n	a	s			
5	a	s					
0	b	a	n	a	n	a	s
2	n	a	n	a	s		
4	n	a	s				
6	s						

Finding an occurrence of a pattern (length m) takes $O(m \log n)$ time

Finding all occurrences takes $O(m \log n + \text{occ})$ time

where occ is the number of occurences

This can be further improved to $O(m + \log n + \text{occ})$ time

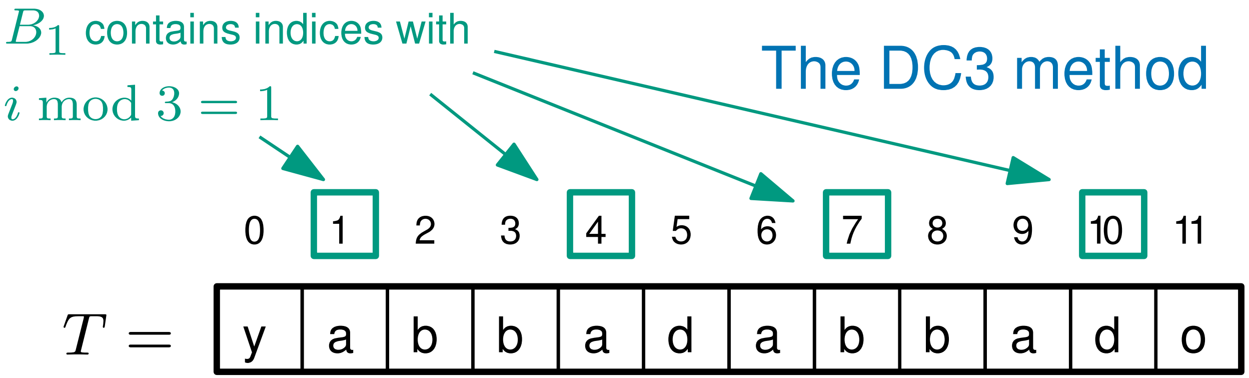
(using LCP queries which we will see in a future lecture)

Do we really need to build the suffix tree to construct the suffix array?

The DC3 method

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

The DC3 method



B_1 contains indices with $i \bmod 3 = 1$

	0	$\boxed{1}$	2	3	$\boxed{4}$	5	6	$\boxed{7}$	8	9	$\boxed{10}$	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

The DC3 method

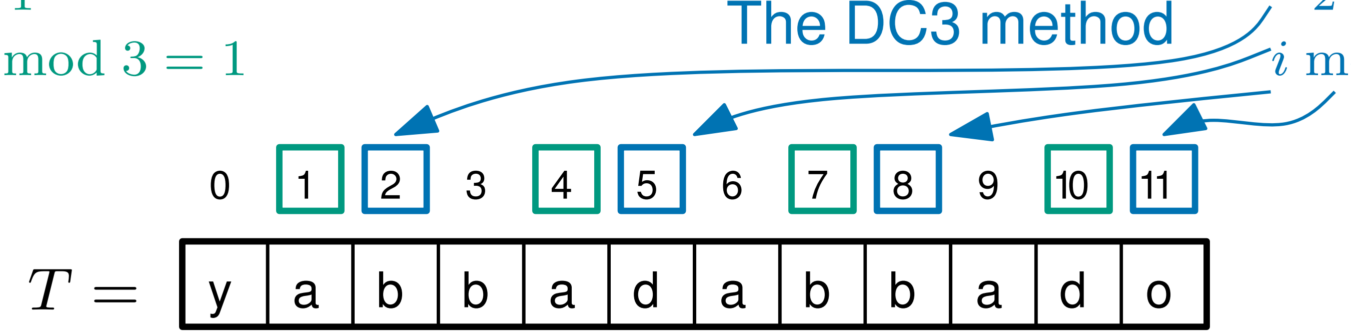
B_1 contains indices with $i \bmod 3 = 1$

	0	$\boxed{1}$	2	3	$\boxed{4}$	5	6	$\boxed{7}$	8	9	$\boxed{10}$	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



The DC3 method

B_1 contains indices with

$i \bmod 3 = 1$

B_2 contains indices with

$i \bmod 3 = 2$

	0	$\boxed{1}$	$\boxed{2}$	3	$\boxed{4}$	$\boxed{5}$	6	$\boxed{7}$	$\boxed{8}$	9	$\boxed{10}$	$\boxed{11}$
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

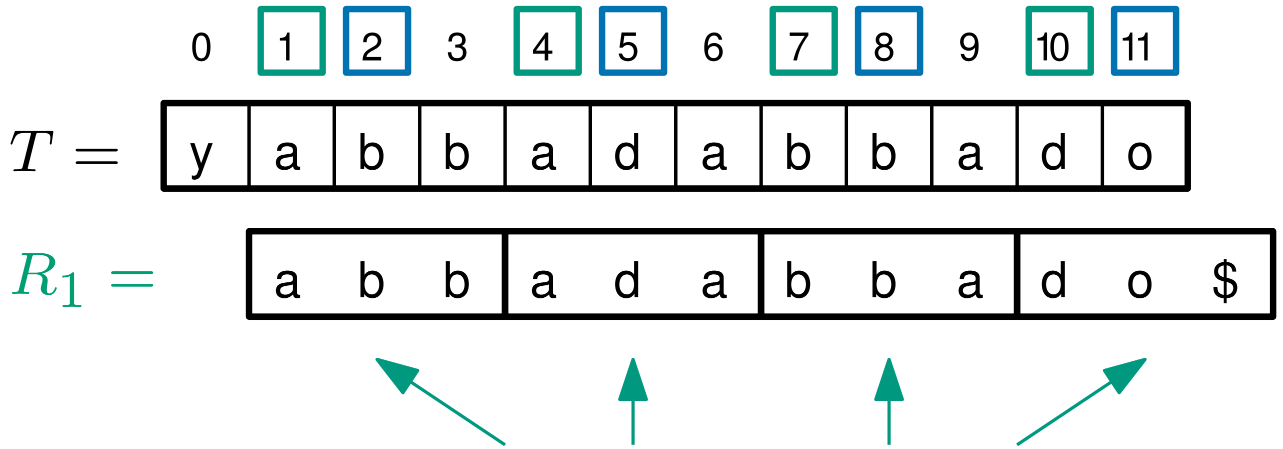
	0	1	2	3	4	5	6	7	8	9	10	11	
$T =$	y	a	b	b	a	d	a	b	b	a	d	o	
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$

Introduce a new “filler symbol” $.

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11	
$T =$	y	a	b	b	a	d	a	b	b	a	d	o	
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$

R_1 is split into *blocks* of length 3

Introduce a new "filler symbol" $.

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

$R_1 =$	a b b	a d a	b b a	d o $

Introduce a new "filler symbol" $.

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new “filler symbol” $.

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

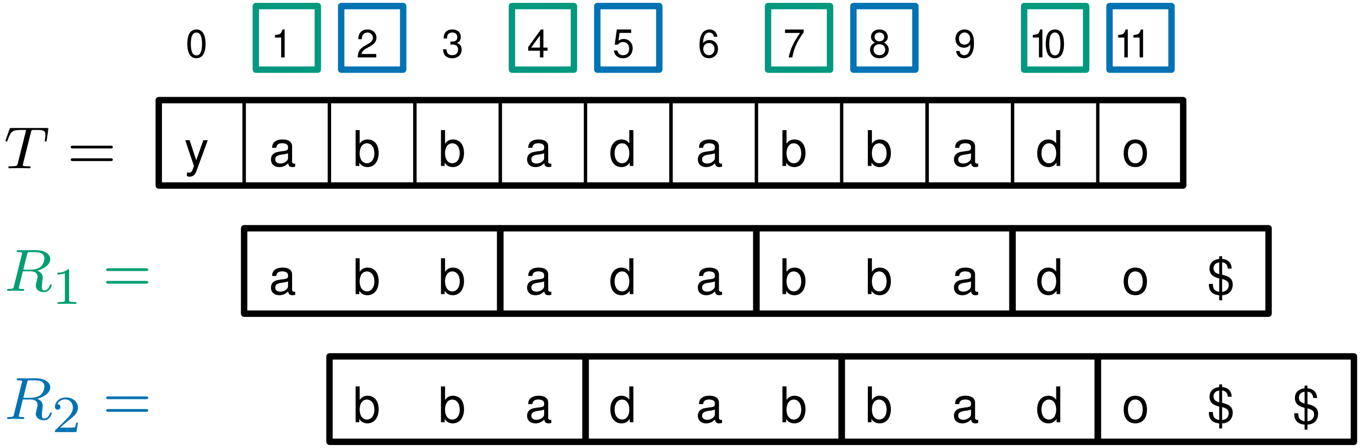
Introduce a new “filler symbol” $.

R_2 is also split into *blocks* of length 3

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



0 1 2 3 4 5 6 7 8 9 10 11

$T =$ y a b b a d a b b a d o

$R_1 =$ a b b | a d a | b b a | d o $

$R_2 =$ b b a | d a b | b a d | o $ $

Introduce a new "filler symbol" $.

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new

“filler symbol” $.

Concatenate R_1 and R_2 to obtain R:

a	b	b	a	d	a	b	b	a	d	o	$	b	b	a	d	a	b	b	a	d	o	$	$

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	\$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	\$	\$

Introduce a new "filler symbol" \$.

Concatenate R_1 and R_2 to obtain R:

a	b	b	a	d	a	b	b	a	d	o	\$	b	b	a	d	a	b	b	a	d	o	\$	\$

Number the blocks in lexicographical order

(\$ is the smallest symbol)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

$R_1 =$ | a b b | a d a | b b a | d o $ |

$R_2 =$ | b b a | d a b | b a d | o $ $ |

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1							

Number the blocks in lexicographical order

($ is the smallest symbol)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new “filler symbol” $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2						

Number the blocks in lexicographical order

($ is the smallest symbol)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

$R_1 =$	a b b	a d a	b b a	d o $
$R_2 =$	b b a	d a b	b a d	o $ $

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2					3	

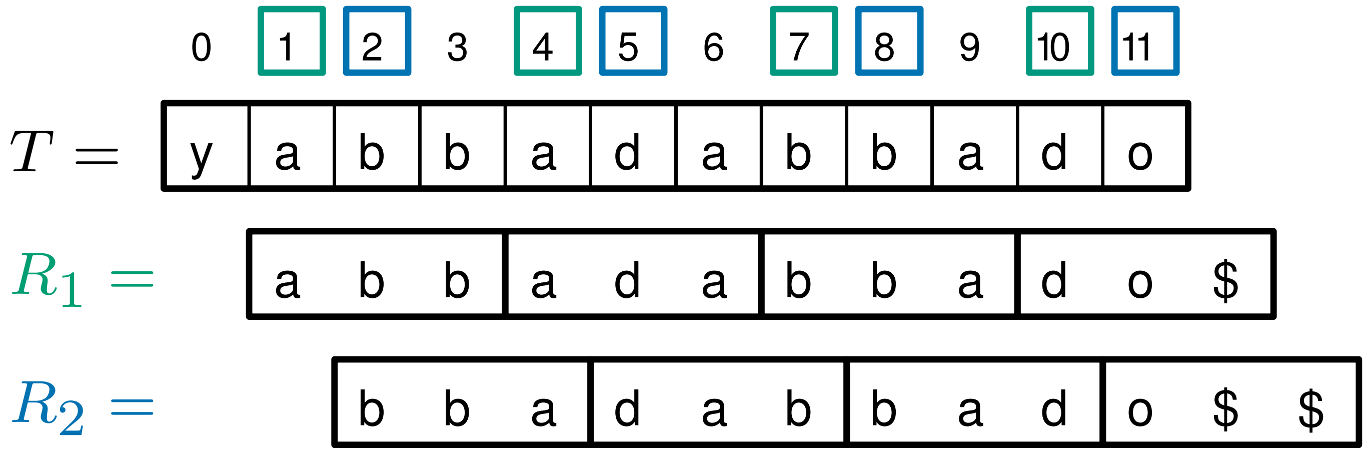
Number the blocks in lexicographical order

($ is the smallest symbol)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



0	1	2	3	4	5	6	7	8	9	10	11

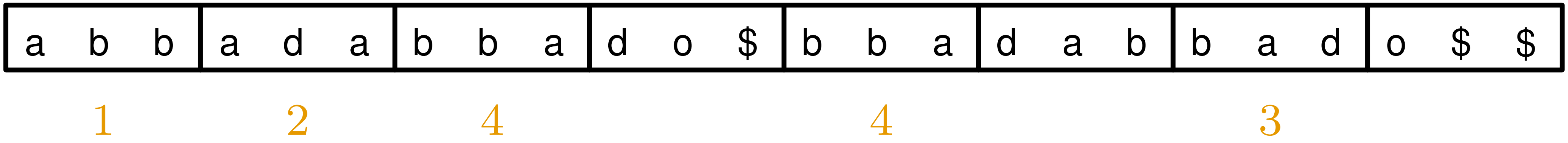
$T =$ y a b b a d a b b a d o

$R_1 =$ a b b | a d a | b b a | d o $

$R_2 =$ b b a | d a b | b a d | o $ $

Introduce a new “filler symbol” $.

Concatenate R_1 and R_2 to obtain R:



a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4		4		3	

Number the blocks in lexicographical order

($ is the smallest symbol)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

$T =$

0	1	2	3	4	5	6	7	8	9	10	11
y	a	b	b	a	d	a	b	b	a	d	o

$R_1 =$

a b b	a d a	b b a	d o $

$R_2 =$

b b a	d a b	b a d	o $ $

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4		4	5	3	

Number the blocks in lexicographical order

($ is the smallest symbol)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11	
$T =$	y	a	b	b	a	d	a	b	b	a	d	o	
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$

$R_2 =$	b b a	d a b	b a d	o $ $

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	

Number the blocks in lexicographical order

($ is the smallest symbol)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new “filler symbol” $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

This can be done by sorting the blocks in $O(n)$ time using radix sort

we assume that the bit representation of each symbol uses $O(\log n)$ bits.

(which is a common and realistic assumption)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

let $R' =$

1	2	4	6	4	5	3	7

The DC3 method

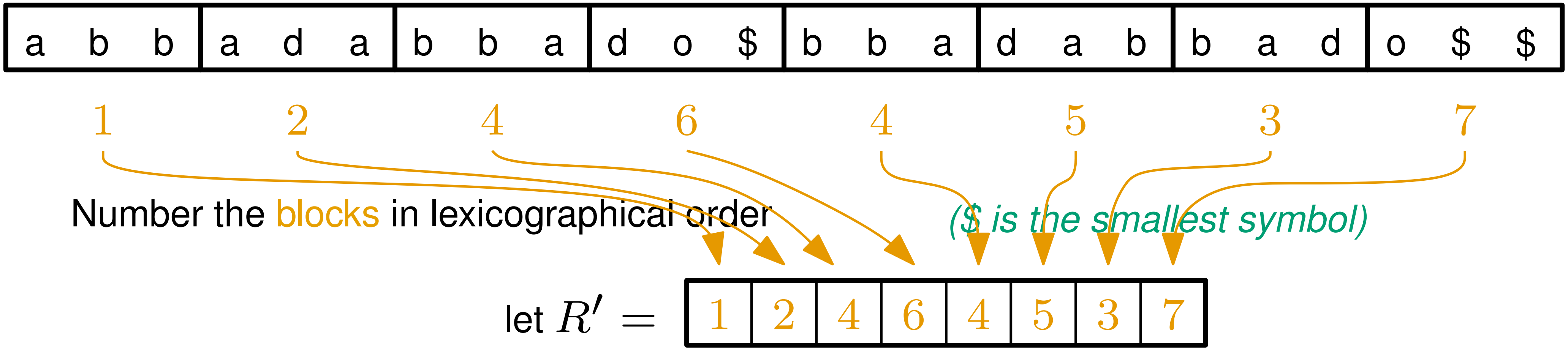
B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:



a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

let $R' =$

1	2	4	6	4	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

let $R' =$

1	2	4	6	4	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new “filler symbol” $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

let $R' =$

0	1	2	3	4	5	6	7
1	2	4	6	4	5	3	7

compute the suffix array of R':

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

$R_1 =$	a b b	a d a	b b a	d o $
$R_2 =$	b b a	d a b	b a d	o $ $

Introduce a new “filler symbol” $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

let $R' =$	1	2	4	6	4	5	3	7
	0	1	2	3	4	5	6	7

compute the suffix array of R':	0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

$R_1 =$	a b b	a d a	b b a	d o $
$R_2 =$	b b a	d a b	b a d	o $ $

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

N

How do we compute the suffix array for R'?

Recursion! *(Notice that R' has length $2n/3$)*

($ is the smallest symbol)

			6	4	5	3	7
0	1	2	3	4	5	6	7

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

$R_1 =$	a b b	a d a	b b a	d o \$
$R_2 =$	b b a	d a b	b a d	o \$ \$

Introduce a new "filler symbol" \$.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

(\$ is the smallest symbol)

let $R' =$

1	2	4	6	4	5	3	7
0	1	2	3	4	5	6	7

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

$R_1 =$	a b b	a d a	b b a	d o $

$R_2 =$	b b a	d a b	b a d	o $ $

Introduce a new "filler symbol" $.

Concatenate R_1 and R_2 to obtain R:

a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

let $R' =$

1	2	4	6	4	5	3	7
0	1	2	3	4	5	6	7

what use is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11		
$T =$	y	a	b	b	a	d	a	b	b	a	d	o		
$R_1 =$		a	b	b	a	d	a	b	b	a	d	o	$	
$R_2 =$			b	b	a	d	a	b	b	a	d	o	$	$

Introduce a new “filler symbol” $.

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
1	2	4	6	4	5	3	7

Number the blocks in lexicographical order

($ is the smallest symbol)

let $R' =$

1	2	4	6	4	5	3	7
0	1	2	3	4	5	6	7

compute the suffix array of R':

0	1	6	4	2	5	3	7

what use

is this?!

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

$T =$

0	1	2	3	4	5	6	7	8	9	10	11
y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

what use is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$

what use

is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
1		a	b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$

what use is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

1	a	b	b	a	d	a	b	b	a	d	o
5	d	a	b	b	a	d	o				

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

what use

is this?!

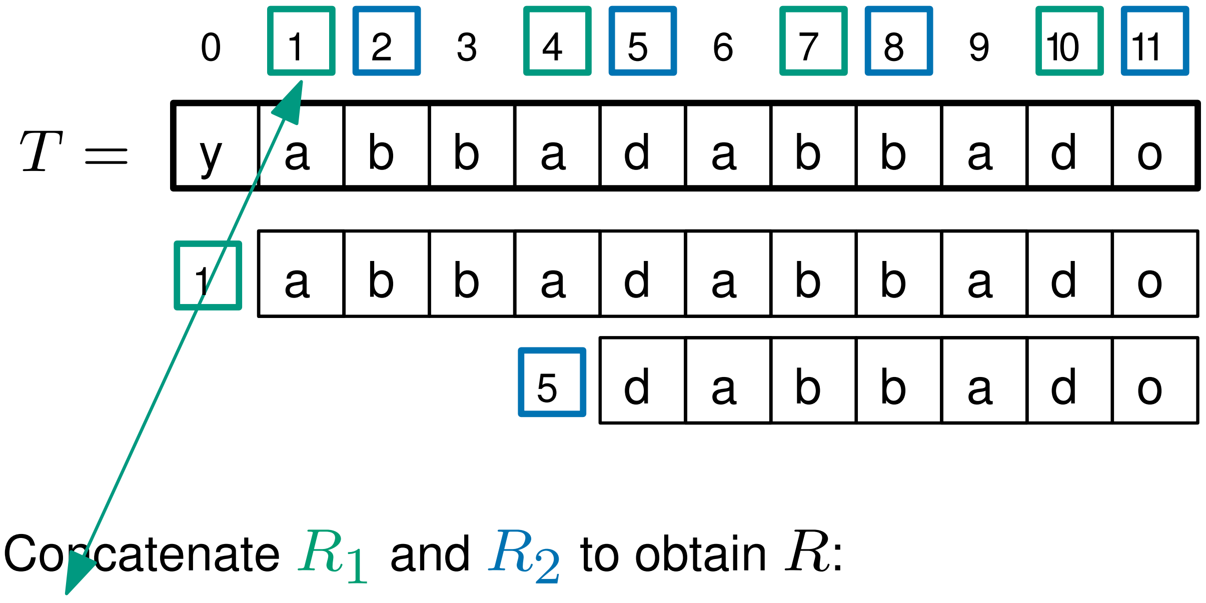
compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

what use

is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
1		a	b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

what use is this?!

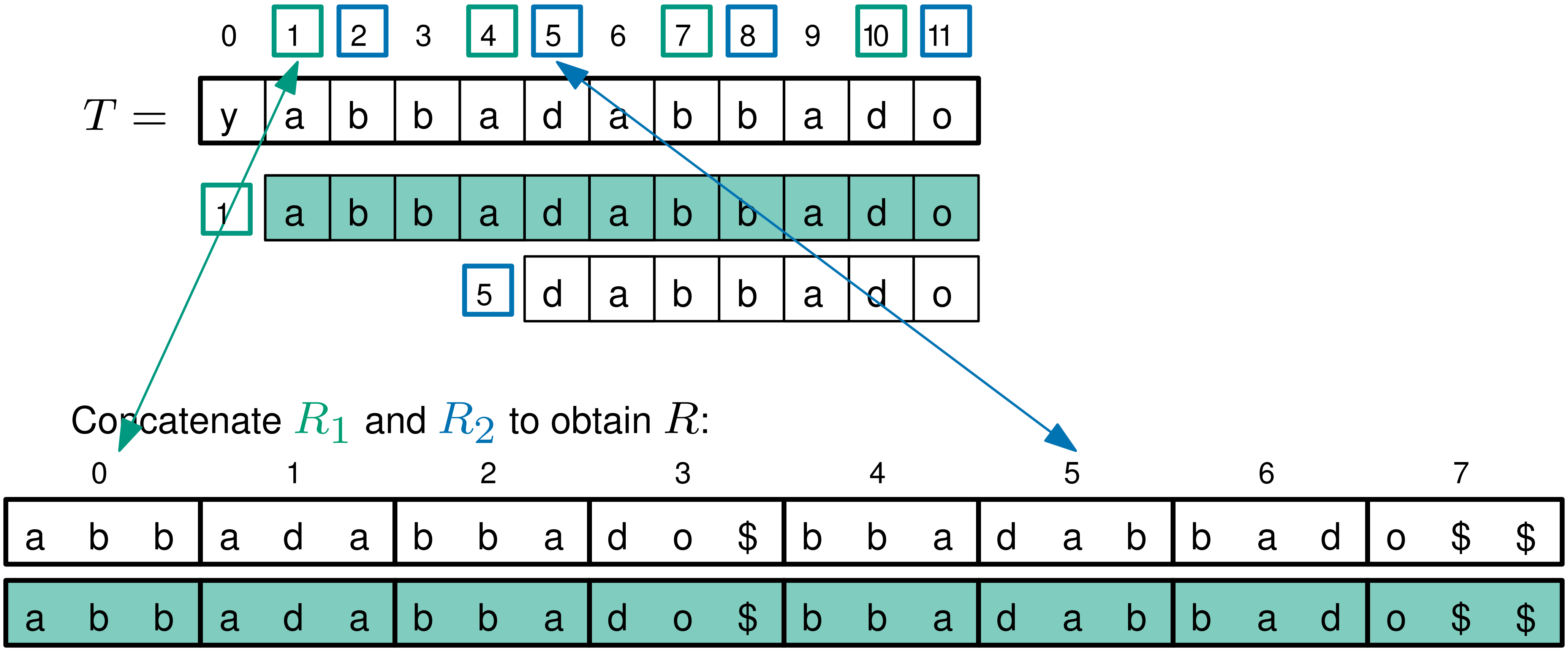
compute the suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
1		a	b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

what use is this?!

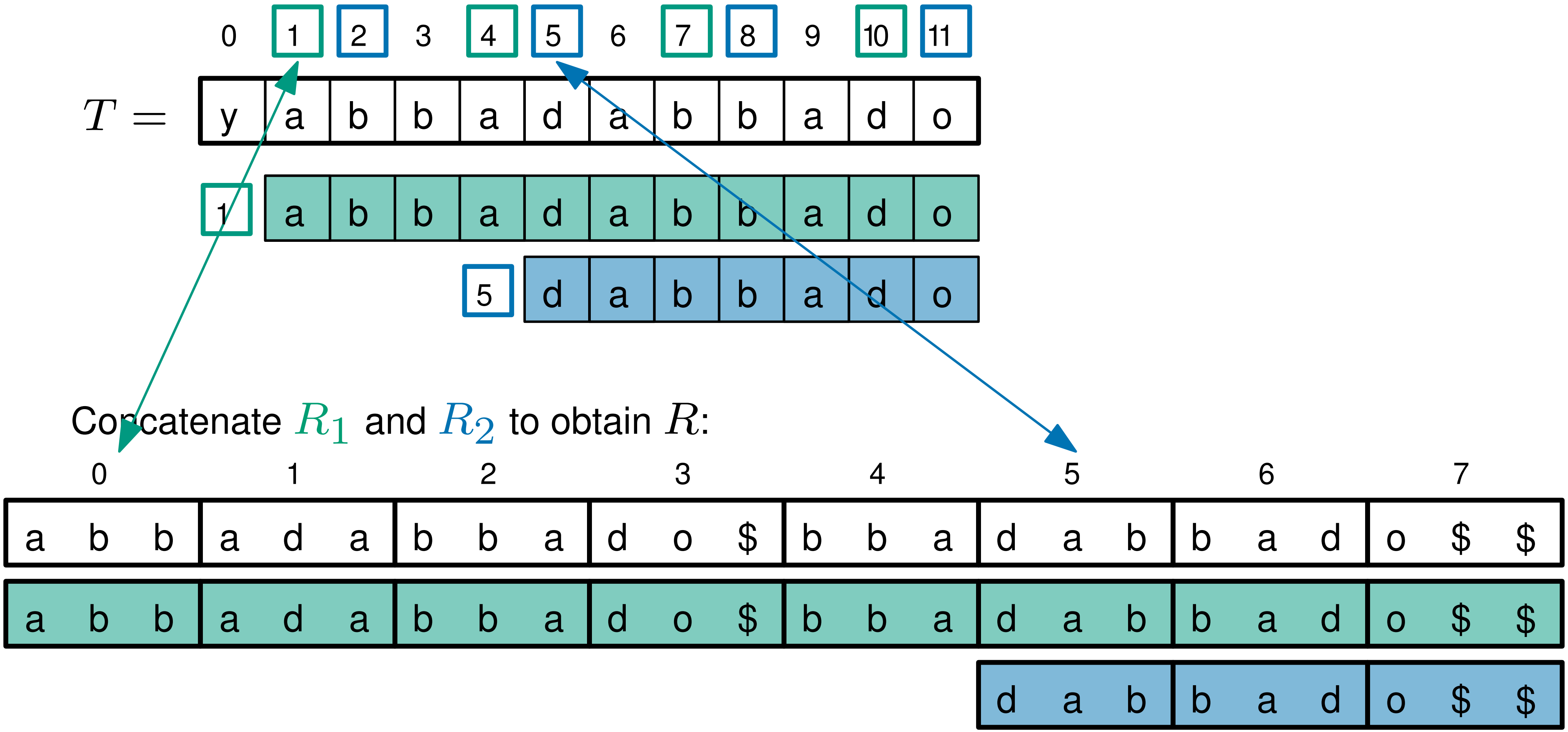
compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
1		a	b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
					d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

what use is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
	1	a	b	b	a	d	a	b	b	a	d	o
					5	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
					d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use

is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
1		a	b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
					d a b	b a d	o $ $

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use

is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
1		a	b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Suffix 1 is smaller than suffix 5

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
					d a b	b a d	o $ $

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use

is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

7			b	b	a	d	o
5	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use is this?!

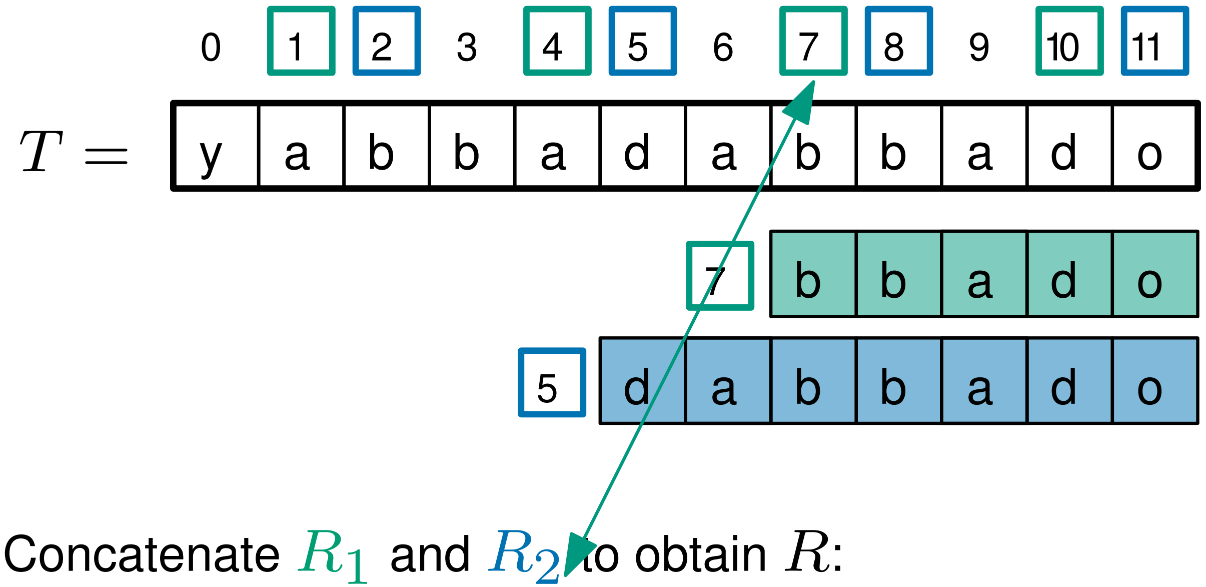
compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



$T =$

0	1	2	3	4	5	6	7	8	9	10	11
y	a	b	b	a	d	a	b	b	a	d	o

7	b	b	a	d	o

5	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
		b b a	d o \$	b b a	d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use is this?!

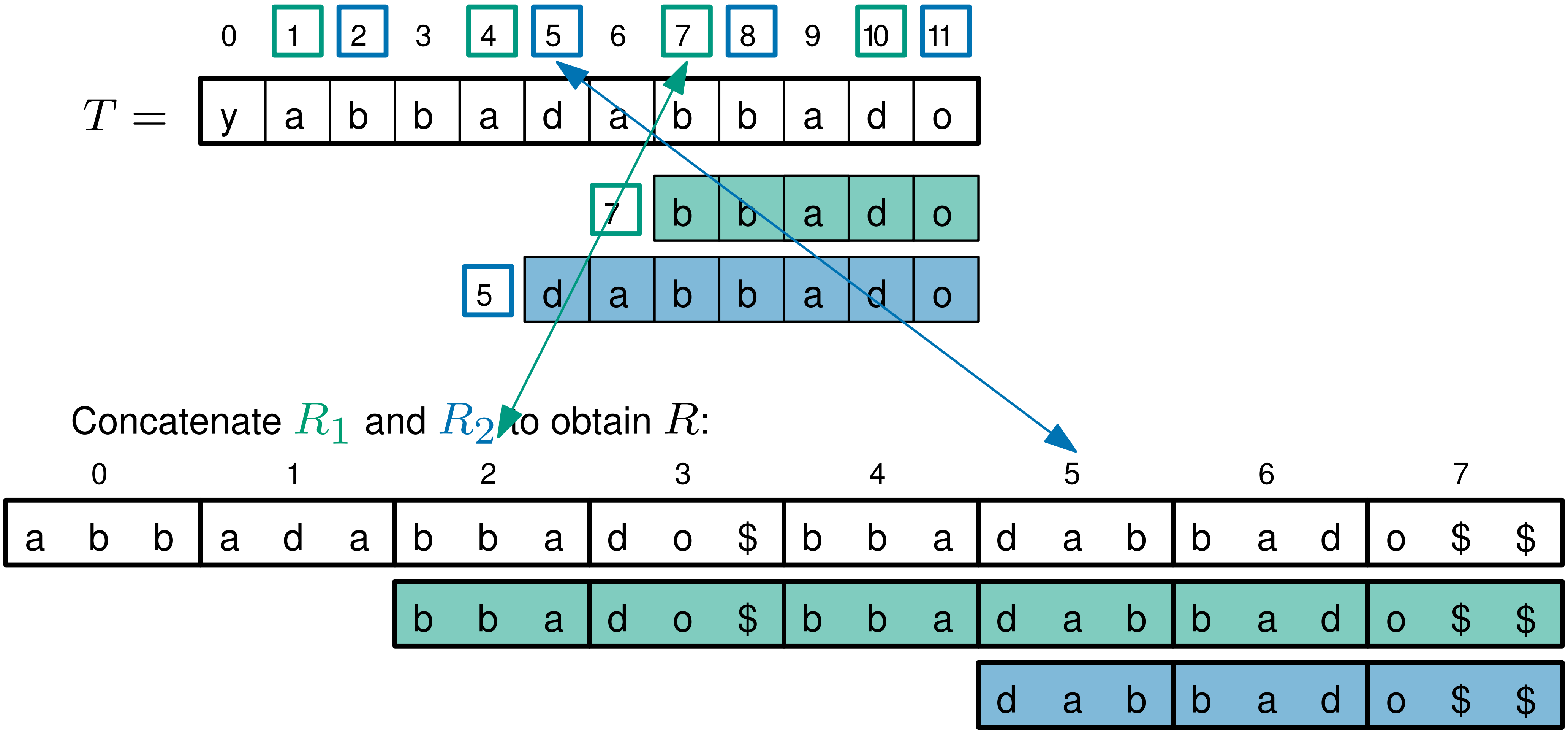
compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

7	b	b	a	d	o

5	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
		b b a	d o \$	b b a	d a b	b a d	o \$ \$
					d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use

is this?!

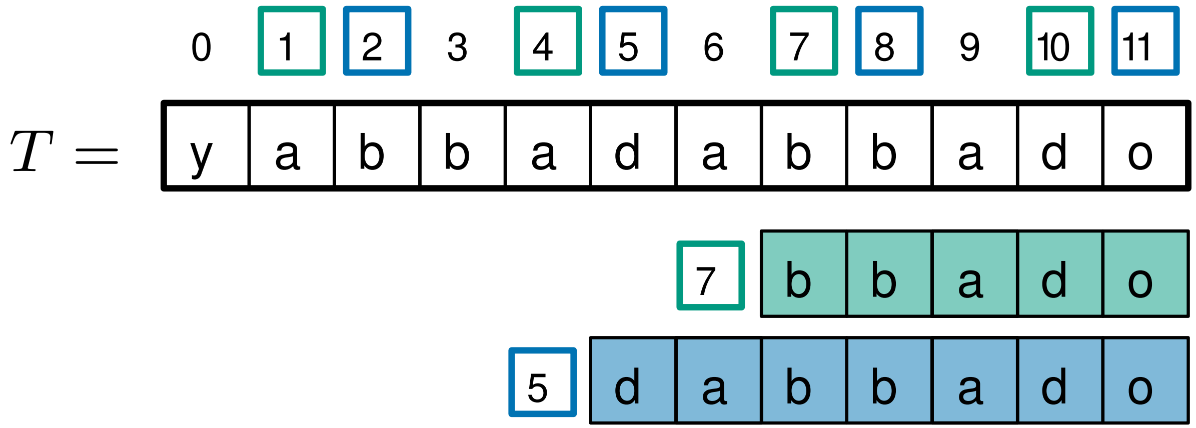
compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

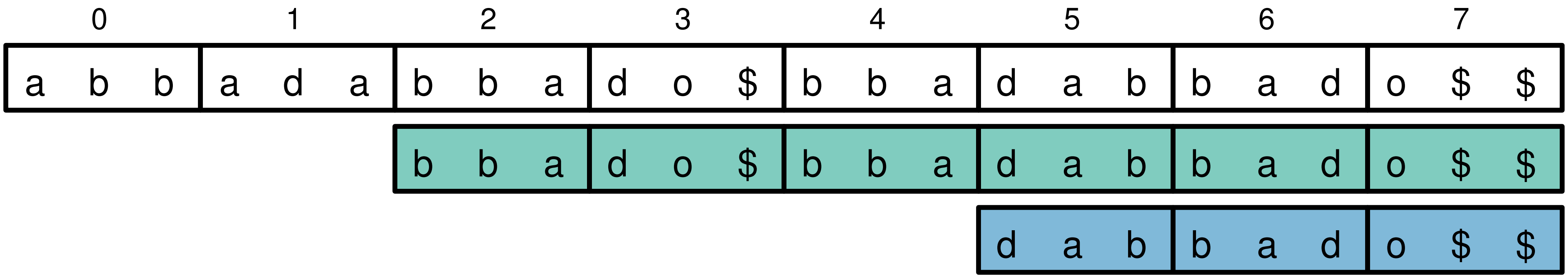


	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

7: b b a d o

5: d a b b a d o

Concatenate R_1 and R_2 to obtain R:



0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
		b b a	d o \$	b b a	d a b	b a d	o \$ \$
					d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use

is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
							7	b	b	a	d	o
					5	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
		b b a	d o \$	b b a	d a b	b a d	o \$ \$
					d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use

is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
7								b	b	a	d	o
5						d	a	b	b	a	d	o

Suffix 7 is smaller than suffix 5

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
		b b a	d o $	b b a	d a b	b a d	o $ $
					d a b	b a d	o $ $

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use

is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use

is this?!

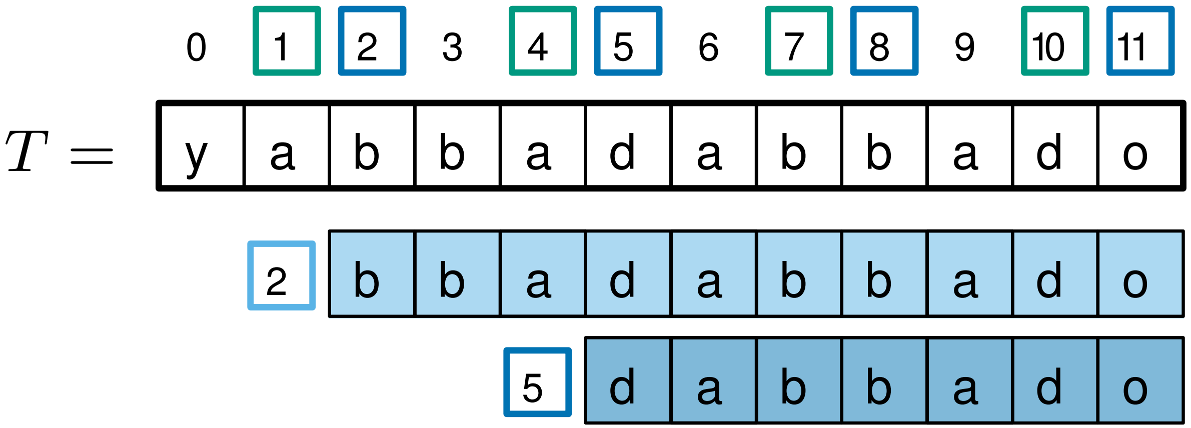
compute the suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

2	b	b	a	d	a	b	b	a	d	o

5	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use

is this?!

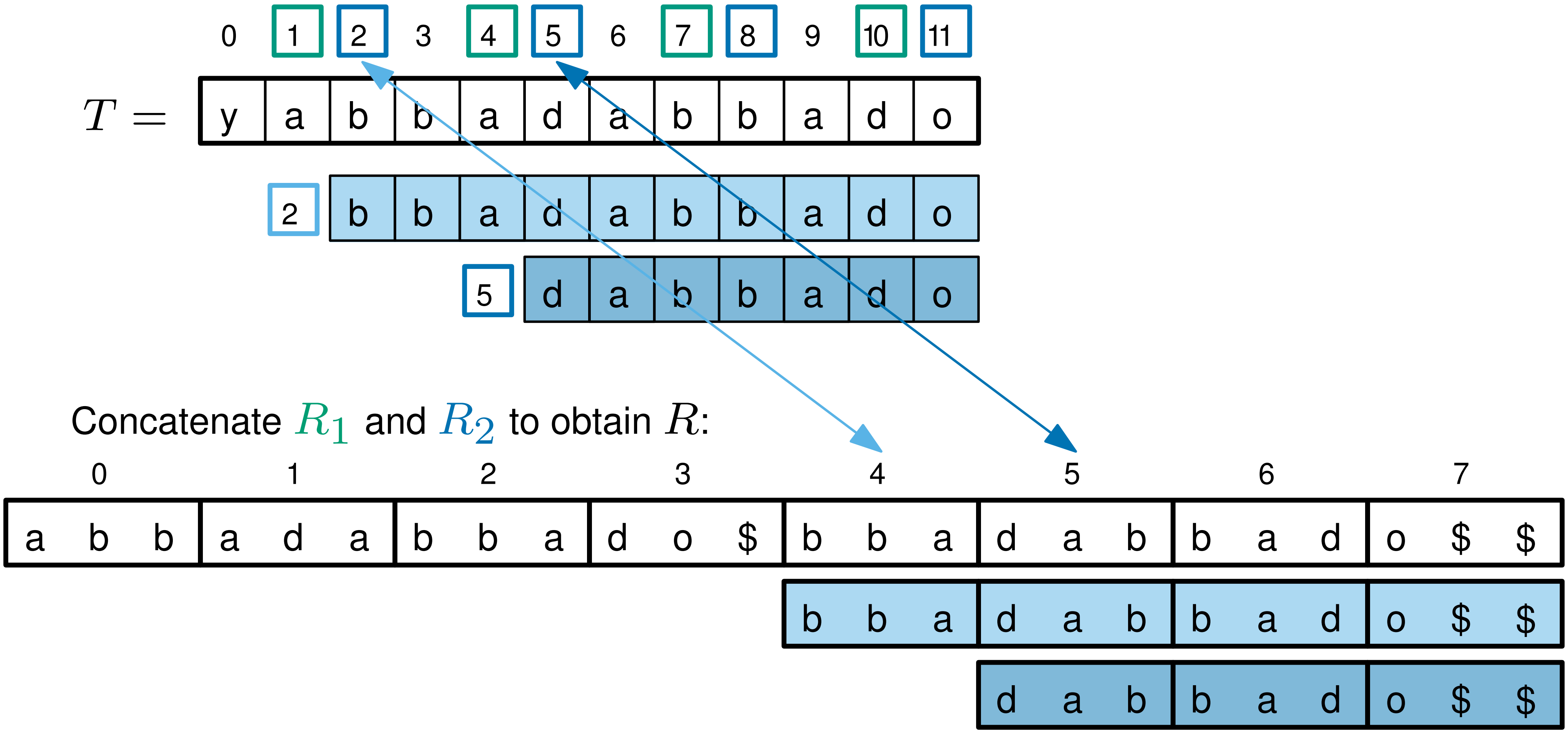
compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
2			b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
				b b a	d a b	b a d	o \$ \$
					d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
2			b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
				b b a	d a b	b a d	o \$ \$
					d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
2			b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$
				b b a	d a b	b a d	o \$ \$
					d a b	b a d	o \$ \$

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o
2			b	b	a	d	a	b	b	a	d	o
5						d	a	b	b	a	d	o

Suffix 2 is smaller than suffix 5

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $
				b b a	d a b	b a d	o $ $
					d a b	b a d	o $ $

Take any two suffixes in $B_1 \cup B_2$ *and find them in* R

their order is given by the suffix array of R':

what use is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

what use

is this?!

compute the suffix array of R':

0	1	6	4	2	5	3	7

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

$T =$

0	$\boxed{1}$	$\boxed{2}$	3	$\boxed{4}$	$\boxed{5}$	6	$\boxed{7}$	$\boxed{8}$	9	$\boxed{10}$	$\boxed{11}$
y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

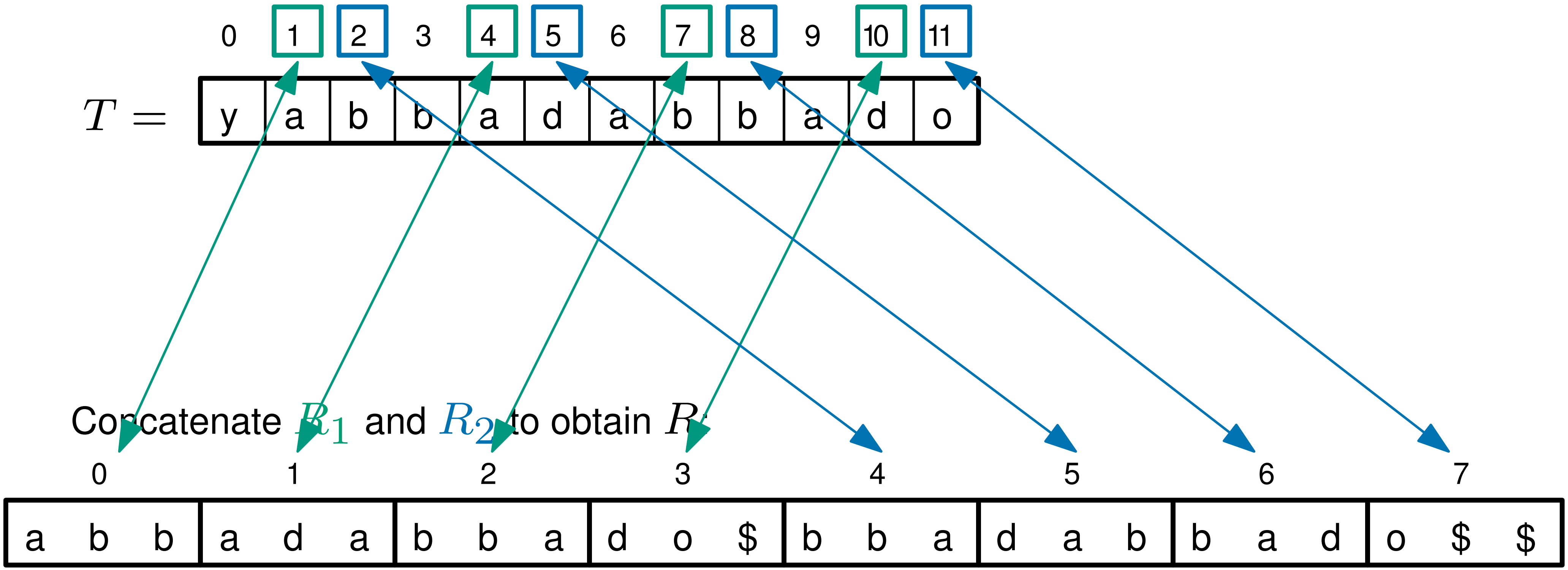
The suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



0	1	2	3	4	5	6	7	8	9	10	11
y	a	b	b	a	d	a	b	b	a	d	o

$T =$

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$

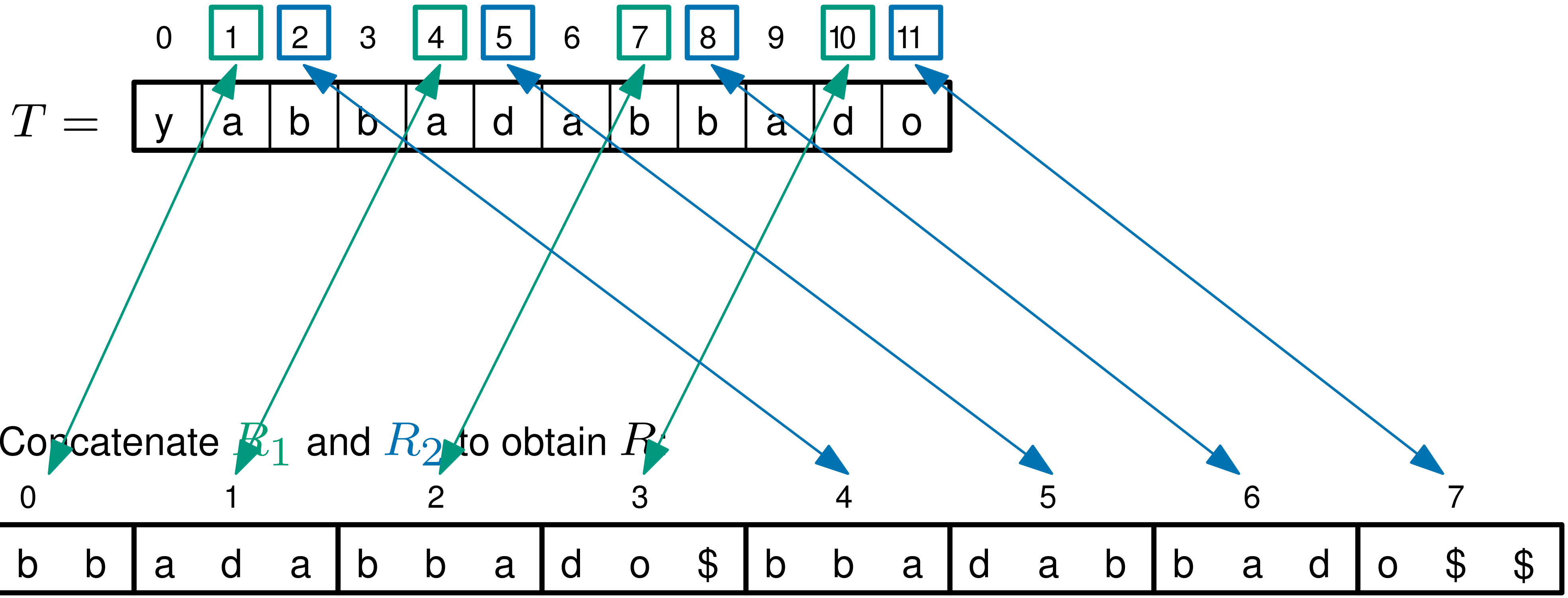
The suffix array of R':

0	1	6	4	2	5	3	7

The DC3 method

B_1 contains indices with
$i \bmod 3 = 1$

B_2 contains indices with
$i \bmod 3 = 2$



0	1	2	3	4	5	6	7	8	9	10	11
y	a	b	b	a	d	a	b	b	a	d	o

$T =$

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o \$	b b a	d a b	b a d	o \$ \$

The suffix array of R':

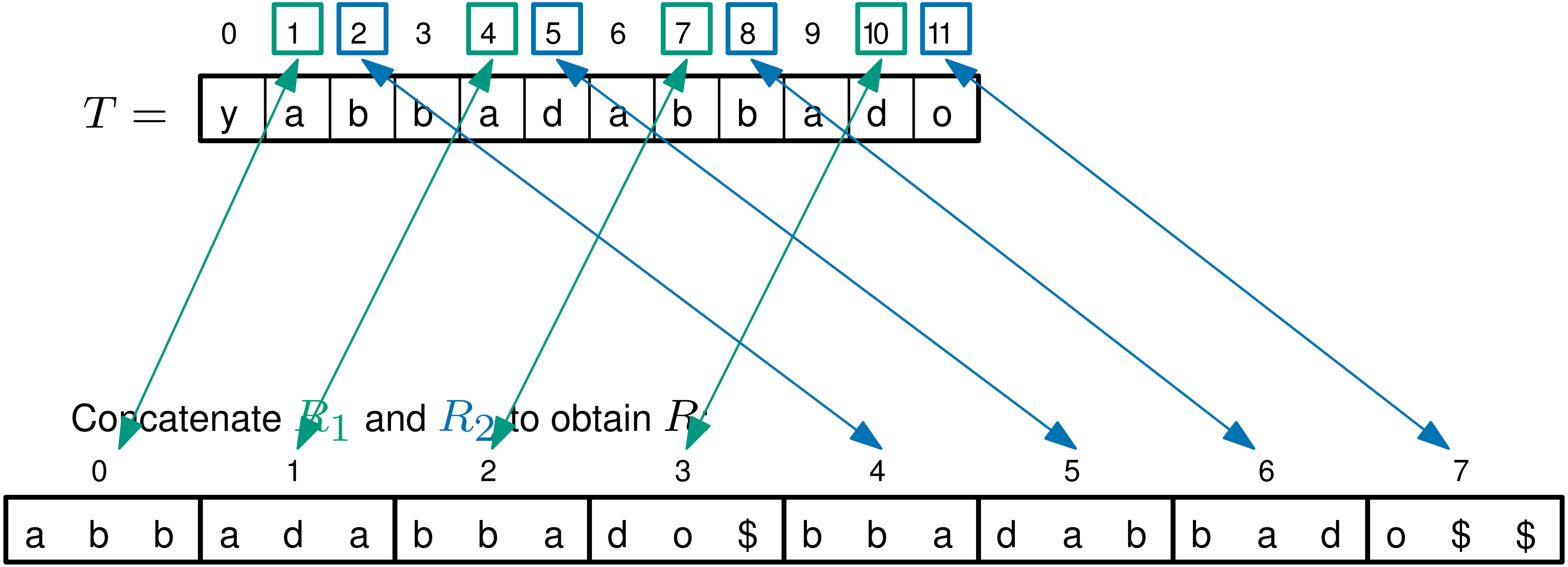
0	1	6	4	2	5	3	7

after relabelling,

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



$T =$

0	1	2	3	4	5	6	7	8	9	10	11
y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

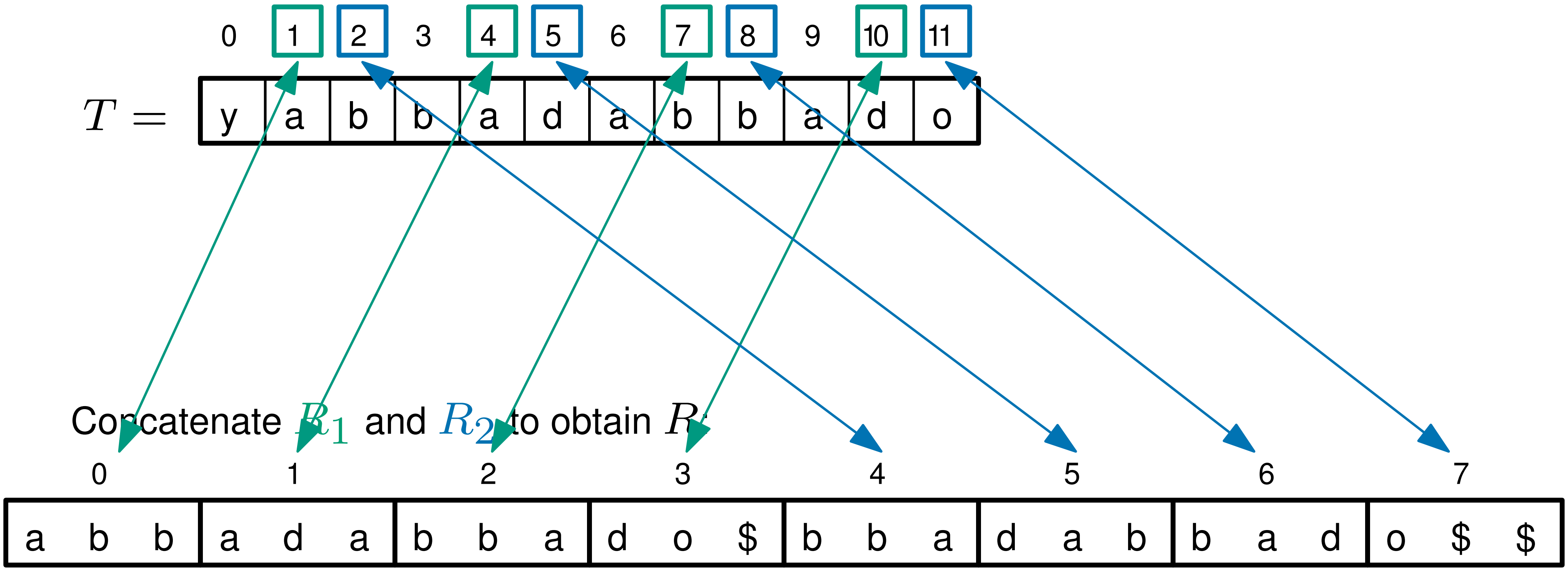
0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

The suffix array of R':	0	1	6	4	2	5	3	7
after relabelling,	1	4	8	2	7	5	10	11

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

The suffix array of R':

0	1	6	4	2	5	3	7

after relabelling,

1	4	8	2	7	5	10	11

we have the suffix array of just the suffixes from $B_1 \cup B_2$

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

The suffix array of R':

0	1	6	4	2	5	3	7

after relabelling,

1	4	8	2	7	5	10	11

we have the suffix array of just the suffixes from $B_1 \cup B_2$

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Concatenate R_1 and R_2 to obtain R:

0	1	2	3	4	5	6	7
a b b	a d a	b b a	d o $	b b a	d a b	b a d	o $ $

The suffix array of R':	0	1	6	4	2	5	3	7
after relabelling,	1	4	8	2	7	5	10	11

we have the suffix array of just the suffixes from $B_1 \cup B_2$

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Suffix array for just $B_1 \cup B_2$

1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

The DC3 method

B_1 contains indices with

$i \bmod 3 = 1$

B_2 contains indices with

$i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

0	y	a	b	b	a	d	a	b	b	a	d	o
3				b	a	d	a	b	b	a	d	o
6							a	b	b	a	d	o
9										a	d	o

Suffix array for just $B_1 \cup B_2$

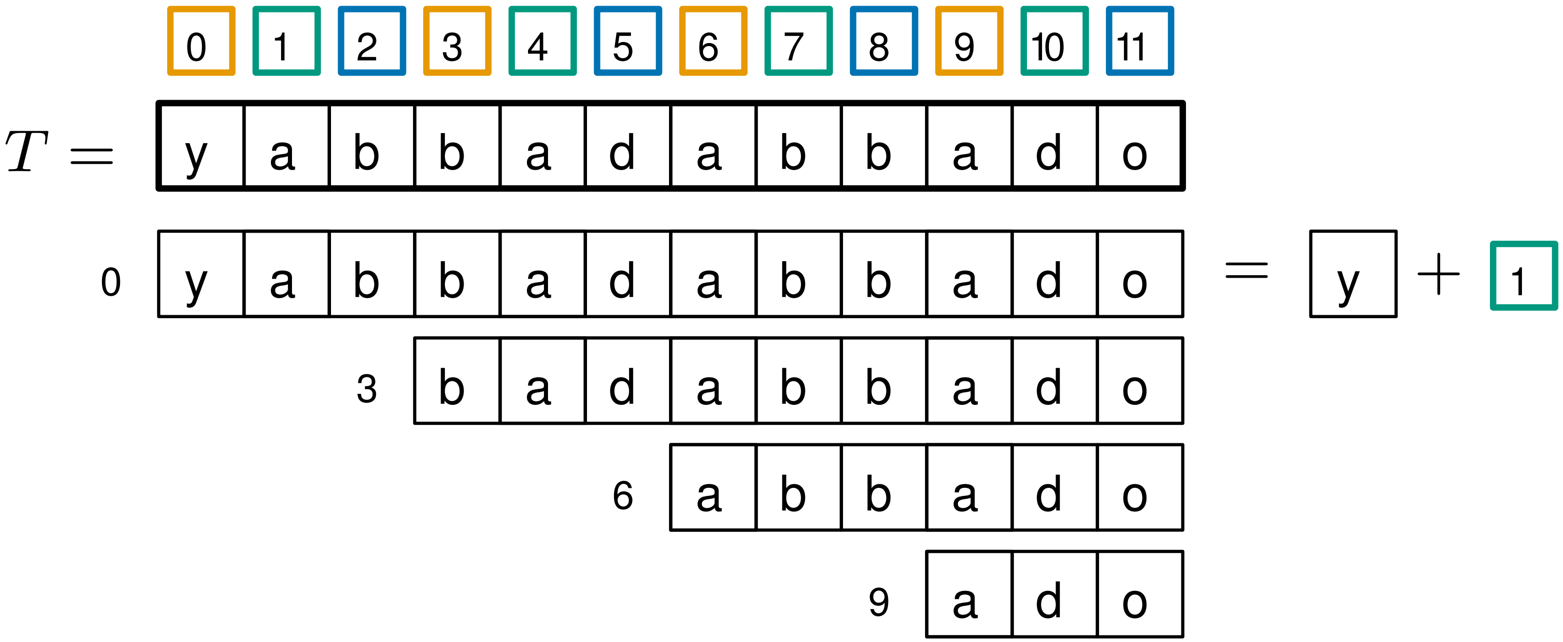
0	1	2	3	4	5	6	7
1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

0 | y | a | b | b | a | d | a | b | b | a | d | o | $=$ y $+$ 1

3 | b | a | d | a | b | b | a | d | o |

6 | a | b | b | a | d | o |

9 | a | d | o |

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

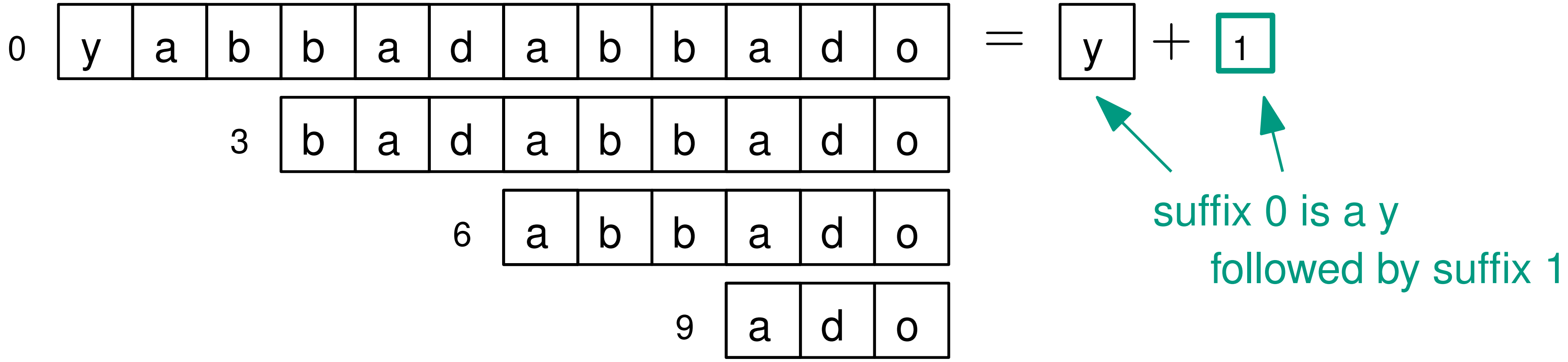
B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

$T =$

0	1	2	3	4	5	6	7	8	9	10	11
y	a	b	b	a	d	a	b	b	a	d	o



0 y a b b a d a b b a d o $=$ y $+$ 1

3 b a d a b b a d o

6 a b b a d o

9 a d o

suffix 0 is a y followed by suffix 1

Suffix array for just $B_1 \cup B_2$

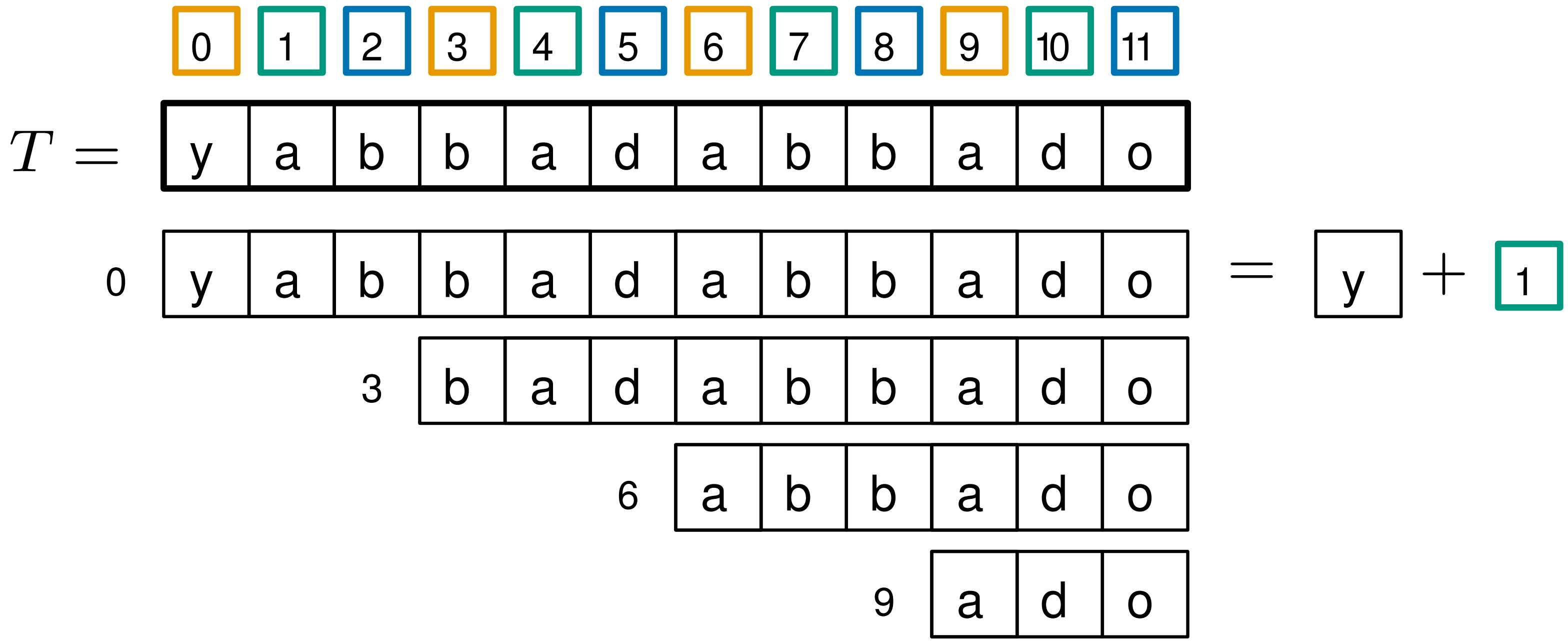
0	1	2	3	4	5	6	7
1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

0 y a b b a d a b b a d o $=$ y $+$ 1

3 b a d a b b a d o

6 a b b a d o

9 a d o

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

0 y a b b a d a b b a d o $=$ y $+$ 1

3 b a d a b b a d o $=$ b $+$ 4

6 a b b a d o $=$ a $+$ 7

9 a d o $=$ a $+$ 10

Suffix array for just $B_1 \cup B_2$

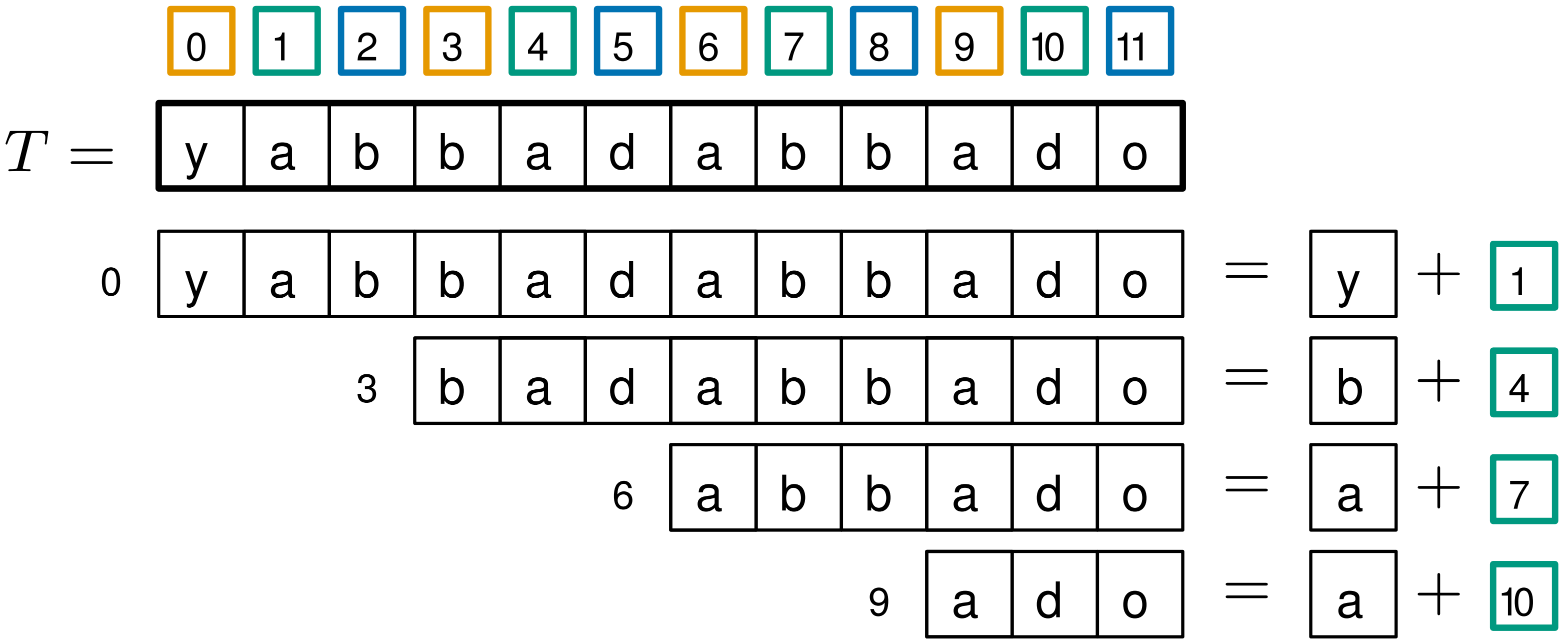
0	1	2	3	4	5	6	7
1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

0 y a b b a d a b b a d o $=$ y $+$ 1

3 b a d a b b a d o $=$ b $+$ 4

6 a b b a d o $=$ a $+$ 7

9 a d o $=$ a $+$ 10

Each suffix $i \in B_0$ is represented by $(T[i], r)$ where r is the rank of suffix $(i+1)$

(the ranks are given by the array below)

Suffix array for just $B_1 \cup B_2$

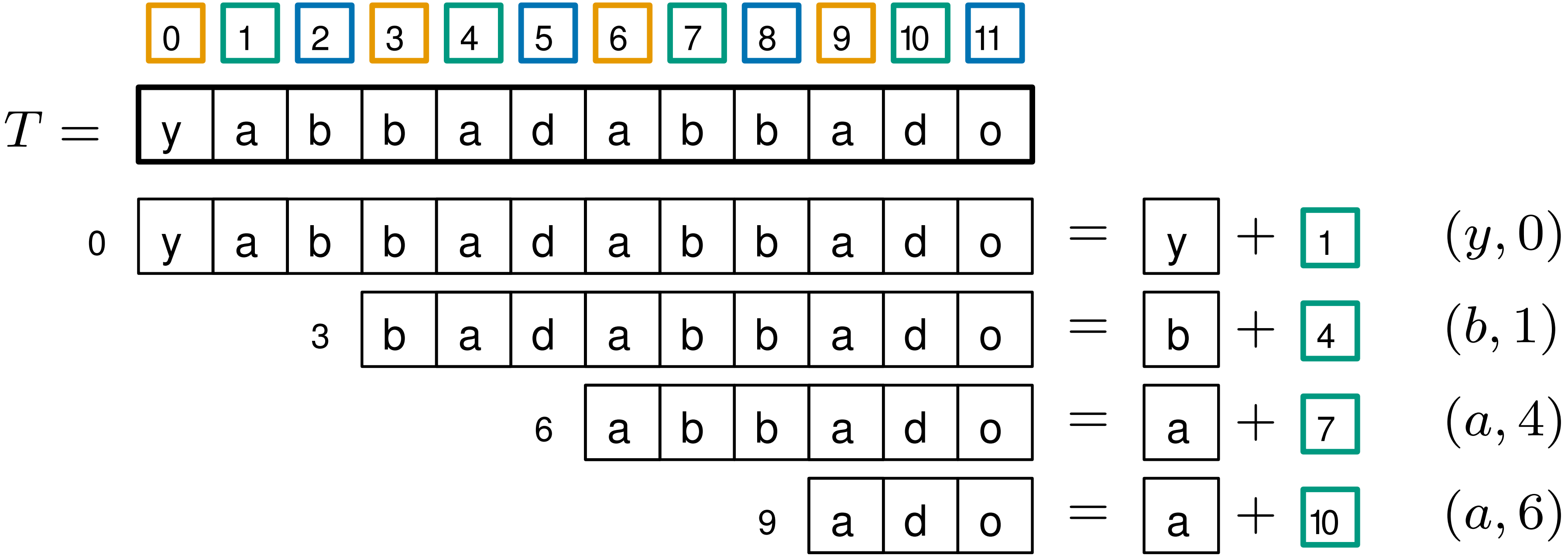
rank:	0	1	2	3	4	5	6	7
	1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

0 y a b b a d a b b a d o $=$ y $+$ 1 $(y, 0)$

3 b a d a b b a d o $=$ b $+$ 4 $(b, 1)$

6 a b b a d o $=$ a $+$ 7 $(a, 4)$

9 a d o $=$ a $+$ 10 $(a, 6)$

Each suffix $i \in B_0$ is represented by $(T[i], r)$ where r is the rank of suffix $(i + 1)$

(the ranks are given by the array below)

Suffix array for just $B_1 \cup B_2$

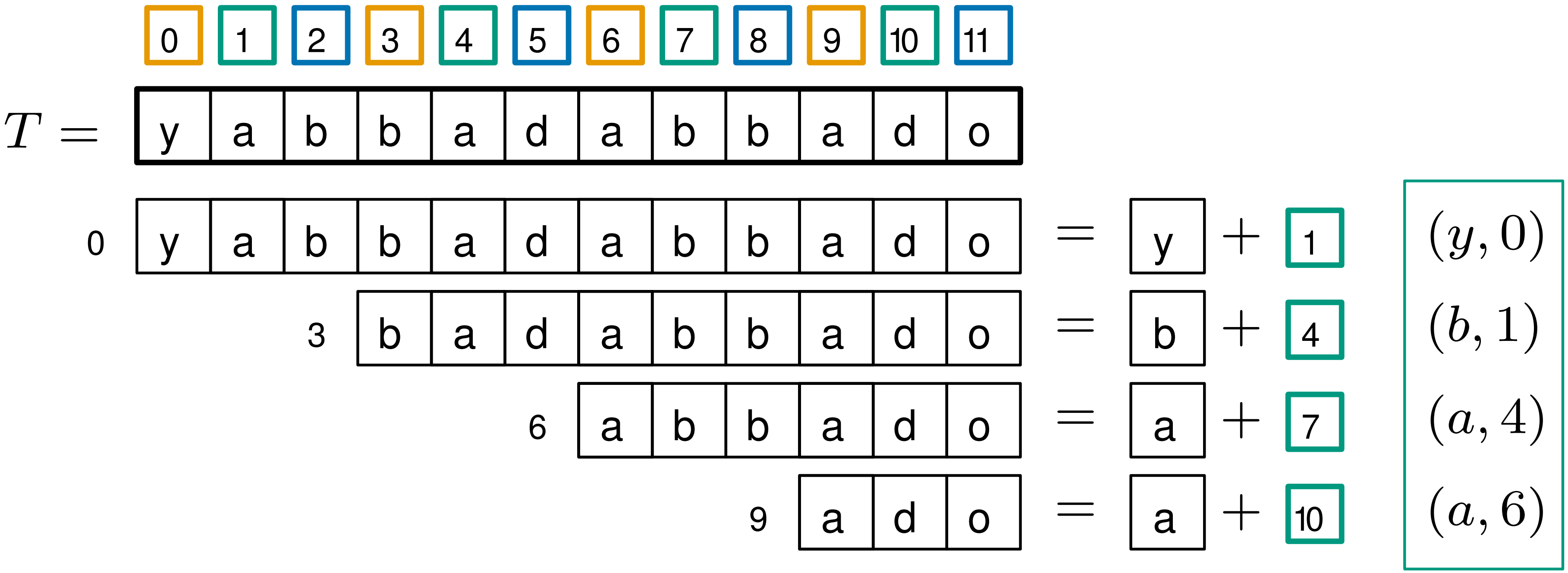
rank:	0	1	2	3	4	5	6	7
	1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



0 1 2 3 4 5 6 7 8 9 10 11

$T =$ y a b b a d a b b a d o

0 y a b b a d a b b a d o = y + 1

3 b a d a b b a d o = b + 4

6 a b b a d o = a + 7

9 a d o = a + 10

$(y, 0)$
$(b, 1)$
$(a, 4)$
$(a, 6)$

Each suffix $i \in B_0$ is represented by $(T[i], r)$ where r is the rank of suffix $(i + 1)$

(the ranks are given by the array below)

We then sort in $O(n)$ time using radix sort

Suffix array for just $B_1 \cup B_2$

rank:	0	1	2	3	4	5	6	7
	1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

$T =$

0	1	2	3	4	5	6	7	8	9	10	11
y	a	b	b	a	d	a	b	b	a	d	o

6 abbado $=$ a $+$ 7 — $(a, 4)$

9 ado $=$ a $+$ 10 — $(a, 6)$

3 badabbado $=$ b $+$ 4 — $(b, 1)$

0 yabbadabbado $=$ y $+$ 1 — $(y, 0)$

Each suffix $i \in B_0$ is represented by $(T[i], r)$ where r is the rank of suffix $(i+1)$

(the ranks are given by the array below)

We then sort in $O(n)$ time using radix sort

Suffix array for just $B_1 \cup B_2$

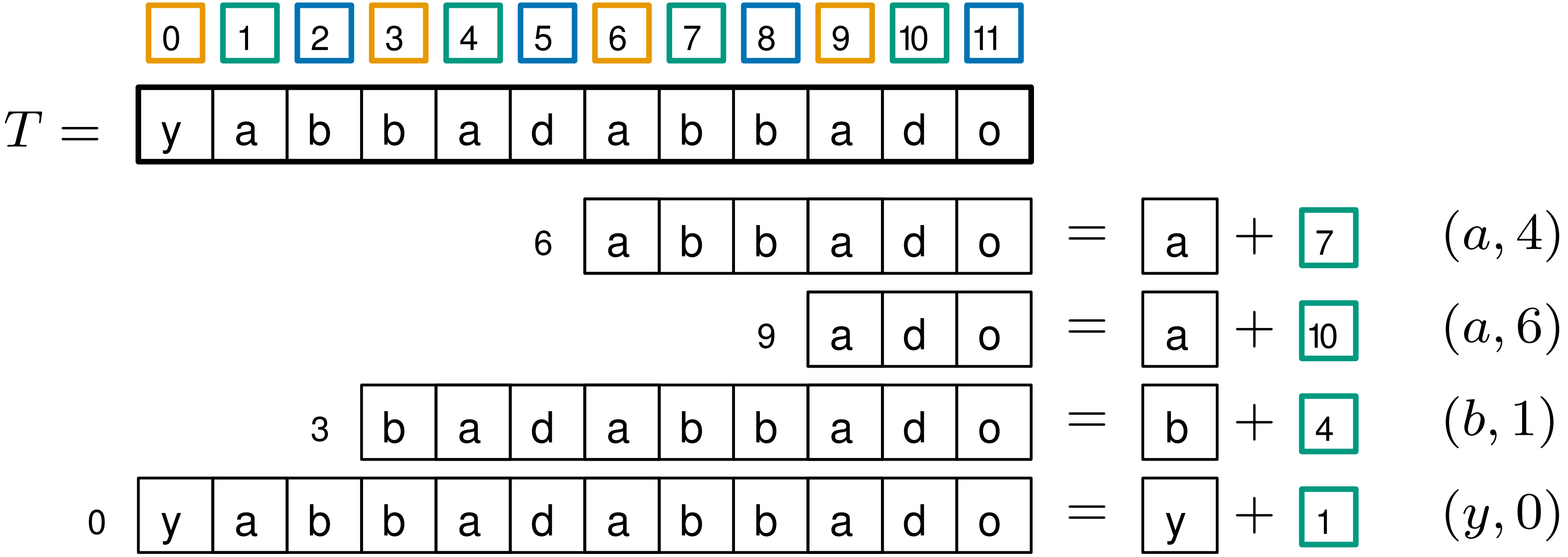
rank:	0	1	2	3	4	5	6	7
	1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

6 a b b a d o $=$ a $+$ 7 $(a, 4)$

9 a d o $=$ a $+$ 10 $(a, 6)$

3 b a d a b b a d o $=$ b $+$ 4 $(b, 1)$

0 y a b b a d a b b a d o $=$ y $+$ 1 $(y, 0)$

Each suffix $i \in B_0$ is represented by $(T[i], r)$ where r is the rank of suffix $(i+1)$

(the ranks are given by the array below)

We then sort in $O(n)$ time using radix sort

Suffix array for just $B_1 \cup B_2$

rank:	0	1	2	3	4	5	6	7
	1	4	8	2	7	5	10	11

How do we find the ordering of the suffixes from B_0? (where $i \bmod 3 = 0$)

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

6 a b b a d o $=$ a $+$ 7 $(a, 4)$

9 a d o $=$ a $+$ 10 $(a, 6)$

3 b a d a b b a d o $=$ b $+$ 4 $(b, 1)$

0 y a b b a d a b b a d o $=$ y $+$ 1 $(y, 0)$

Each suffix $i \in B_0$ is represented by $(T[i], r)$ where r is the rank of suffix $(i + 1)$

(the ranks are given by the array below)

We then sort in $O(n)$ time using radix sort

Suffix array for just $B_1 \cup B_2$

rank:	0	1	2	3	4	5	6	7
	1	4	8	2	7	5	10	11

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

6 abbado $=$ a $+$ $\boxed{7}$ $(a, 4)$

9 ado $=$ a $+$ $\boxed{10}$ $(a, 6)$

3 badabbado $=$ b $+$ $\boxed{4}$ $(b, 1)$

0 yabbadabbado $=$ y $+$ $\boxed{1}$ $(y, 0)$

Each suffix $i \in B_0$ is represented by $(T[i], r)$ where r is the rank of suffix $(i+1)$

(the ranks are given by the array below)

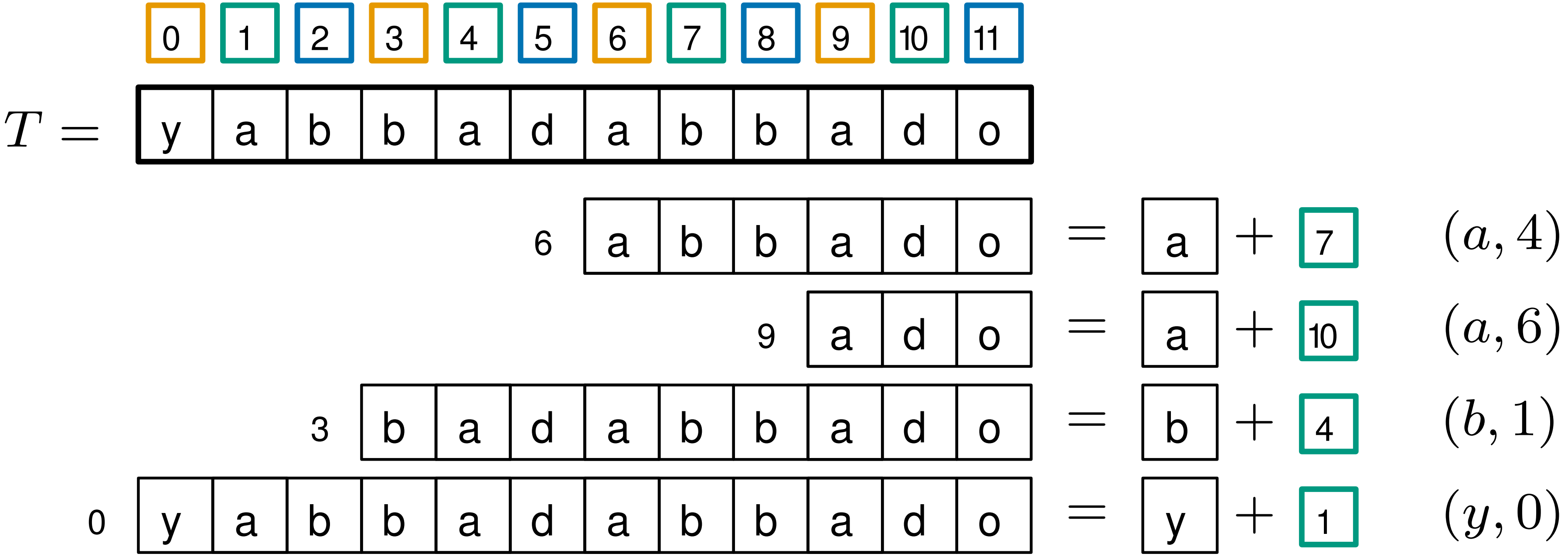
We then sort in $O(n)$ time using radix sort

rank:	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	1	4	8	2	7	5	10	11
Suffix array for just B_0	6	9	3	0				

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$



	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

6 a b b a d o = a + 7 $(a, 4)$

9 a d o = a + 10 $(a, 6)$

3 b a d a b b a d o = b + 4 $(b, 1)$

0 y a b b a d a b b a d o = y + 1 $(y, 0)$

Each suffix $i \in B_0$ is represented by $(T[i], r)$ where r is the rank of suffix $(i+1)$

(the ranks are given by the array below)

We then sort in $O(n)$ time using radix sort

rank:	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	1	4	8	2	7	5	10	11
Suffix array for just B_0	6	9	3	0				

How do we merge these?

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	1	4	8	2	7	5	10	11
Suffix array for just B_0	6	9	3	0				

How do we merge these?

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort...

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	1	4	8	2	7	5	10	11
Suffix array for just B_0	6	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . .

which is smaller, suffix 1 or 6 ?

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	1	4	8	2	7	5	10	11
Suffix array for just B_0	6	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . .

which is smaller, suffix $\boxed{1}$ or $\boxed{6}$?

$$\boxed{6} = \boxed{a} + \boxed{7}$$

$$\boxed{1} = \boxed{a} + \boxed{2}$$

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	1	4	8	2	7	5	10	11
Suffix array for just B_0	6	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . .

which is smaller, suffix $\boxed{1}$ or $\boxed{6}$?

$$\boxed{6} = \boxed{a} + \boxed{7} \qquad (a, 4)$$

$$\boxed{1} = \boxed{a} + \boxed{2} \qquad (a, 3)$$

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	1	4	8	2	7	5	10	11
Suffix array for just B_0	6	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort...

which is smaller, suffix $\boxed{1}$ or $\boxed{6}$?

$$\boxed{6} = \boxed{\text{a}} + \boxed{7} \qquad (a, 4)$$

$$\boxed{1} = \boxed{\text{a}} + \boxed{2} \qquad (a, 3)$$

It takes $O(1)$ time to decide that $\boxed{1}$ is smaller

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
1	4	8	2	7	5	10	11

Suffix array for just B_0

6	9	3	0

How do we merge these?

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . . $\boxed{1}$

which is smaller, suffix $\boxed{1}$ or $\boxed{6}$?

$\boxed{6} = \boxed{a} + \boxed{7}$ $(a, 4)$

$\boxed{1} = \boxed{a} + \boxed{2}$ $(a, 3)$

It takes $O(1)$ time to decide that $\boxed{1}$ is smaller

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
~~1~~	4	8	2	7	5	10	11

Suffix array for just B_0

6	9	3	0

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . . $\boxed{1}$

which is smaller, suffix $\boxed{4}$ or $\boxed{6}$?

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	~~1~~	4	8	2	7	5	10	11
Suffix array for just B_0	6	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . . $\boxed{1}$

which is smaller, suffix $\boxed{4}$ or $\boxed{6}$?

$$\boxed{6} = \boxed{a} + \boxed{7}$$

$$\boxed{4} = \boxed{a} + \boxed{5}$$

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
~~1~~	4	8	2	7	5	10	11

Suffix array for just B_0

6	9	3	0

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . . $\boxed{1}$

which is smaller, suffix $\boxed{4}$ or $\boxed{6}$?

$\boxed{6} = \boxed{a} + \boxed{7} \qquad (a, 4)$

$\boxed{4} = \boxed{a} + \boxed{5} \qquad (a, 5)$

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
~~1~~	4	8	2	7	5	10	11

Suffix array for just B_0

6	9	3	0

How do we merge these?

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . . $\boxed{1}$

which is smaller, suffix $\boxed{4}$ or $\boxed{6}$?

$$\boxed{6} = \boxed{\text{a}} + \boxed{7} \qquad (a, 4)$$

$$\boxed{4} = \boxed{\text{a}} + \boxed{5} \qquad (a, 5)$$

Again, it takes $O(1)$ time to decide that $\boxed{6}$ is smaller

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
~~1~~	4	8	2	7	5	10	11

Suffix array for just B_0

6	9	3	0

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort…

1	6

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	~~1~~	4	8	2	7	5	10	11
Suffix array for just B_0	~~6~~	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . .

1	6

which is smaller, suffix 4 or 9 ?

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	~~1~~	4	8	2	7	5	10	11
Suffix array for just B_0	~~6~~	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort...

1	6

which is smaller, suffix 4 or 9 ? (4 is smaller)

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	~~1~~	4	8	2	7	5	10	11
Suffix array for just B_0	~~6~~	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort...

1	6	4

which is smaller, suffix 4 or 9 ? (4 is smaller)

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	~~1~~	~~4~~	8	2	7	5	10	11
Suffix array for just B_0	~~6~~	9	3	0				

How do we merge these?

The DC3 method

B_1 contains indices with $i \bmod 3 = 1$

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort...

1	6	4

which is smaller, suffix 8 or 9 ?

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
~~1~~	~~4~~	8	2	7	5	10	11

Suffix array for just B_0

~~6~~	9	3	0

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort...

1	6	4

which is smaller, suffix $\boxed{8}$ or $\boxed{9}$?

$$\boxed{9} = \boxed{\text{a}} + \boxed{10}$$

$$\boxed{8} = \boxed{\text{b}} + \boxed{9}$$

Suffix array for just $B_1 \cup B_2$

0	1	2	3	4	5	6	7
~~1~~	~~4~~	8	2	7	5	10	11

Suffix array for just B_0

~~6~~	9	3	0

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort...

1	6	4

which is smaller, suffix $\boxed{8}$ or $\boxed{9}$?

$$\boxed{9} = \boxed{a} + \boxed{10}$$

$$\boxed{8} = \boxed{b} + \boxed{9}$$

Uh oh! how do we compare $\boxed{9}$ to $\boxed{10}$?

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	~~1~~	~~4~~	8	2	7	5	10	11
Suffix array for just B_0	~~6~~	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort. . .

1	6	4

which is smaller, suffix $\boxed{8}$ or $\boxed{9}$?

$$\boxed{9} = \boxed{a} + \boxed{d} + \boxed{11}$$

$$\boxed{8} = \boxed{b} + \boxed{a} + \boxed{10}$$

It *still* takes $O(1)$ time to decide that $\boxed{9}$ is smaller

Uh oh! how do we compare $\boxed{9}$ to $\boxed{10}$?

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	~~1~~	~~4~~	8	2	7	5	10	11
Suffix array for just B_0	~~6~~	9	3	0				

How do we merge these?

B_1 contains indices with $i \bmod 3 = 1$

The DC3 method

B_2 contains indices with $i \bmod 3 = 2$

	0	1	2	3	4	5	6	7	8	9	10	11
$T =$	y	a	b	b	a	d	a	b	b	a	d	o

Merge them like in mergesort...

1	6	4

Overall this merging phase takes $O(n)$ time

(because processing each suffix takes $O(1)$ time)

	0	1	2	3	4	5	6	7
Suffix array for just $B_1 \cup B_2$	~~1~~	~~4~~	8	2	7	5	10	11
Suffix array for just B_0	~~6~~	9	3	0				

How do we merge these?

The DC3 method

Theorem

The DC3 algorithm constructs a suffix array in $O(n)$ time.

The DC3 method

Theorem

The DC3 algorithm constructs a suffix array in $O(n)$ time.

Proof

Suppose $T(n)$ is the running time. We have

$$T(n) = T(2n/3) + O(n)$$

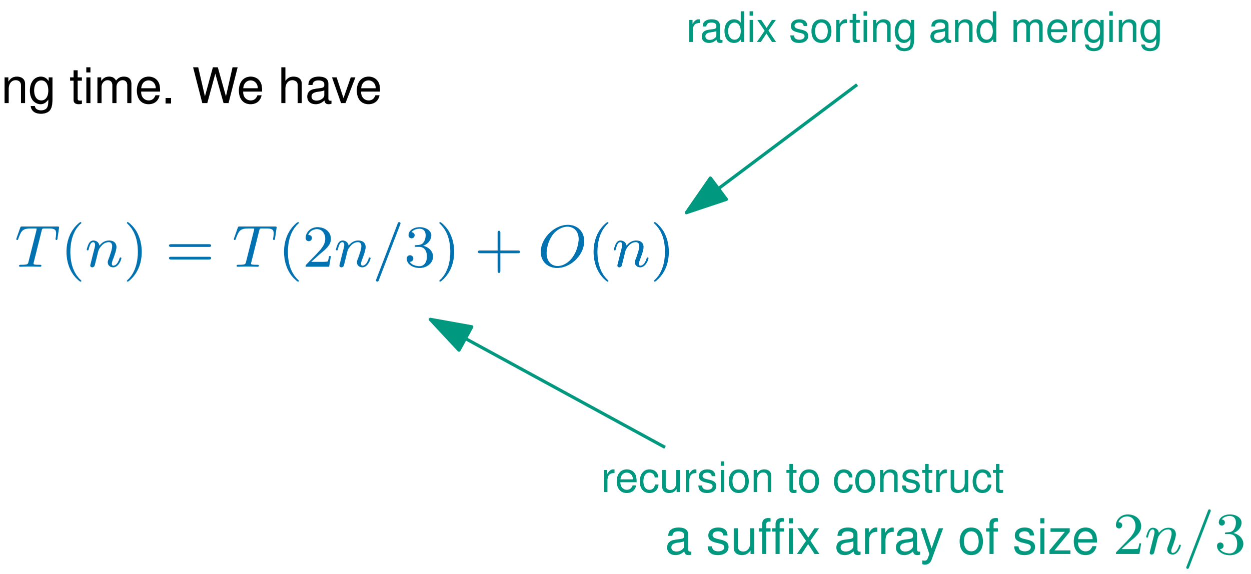
The DC3 method

Theorem

The DC3 algorithm constructs a suffix array in $O(n)$ time.

Proof

Suppose $T(n)$ is the running time. We have



radix sorting and merging

$$T(n) = T(2n/3) + O(n)$$

recursion to construct
a suffix array of size $2n/3$

The DC3 method

Theorem

The DC3 algorithm constructs a suffix array in $O(n)$ time.

Proof

Suppose $T(n)$ is the running time. We have

radix sorting and merging

$$T(n) = T(2n/3) + O(n)$$

recursion to construct a suffix array of size $2n/3$

Solving this recurrence gives $T(n) \in O(n)$.

The suffix array

n

T	b	a	n	a	n	a	s

Sort the suffixes lexicographically

Suffix Array	1	3	5	0	2	4	6

n

P	a	n	a	n	a

m

1	a	n	a	n	a	s	
3	a	n	a	s			
5	a	s					
0	b	a	n	a	n	a	s
2	n	a	n	a	s		
4	n	a	s				
6	s						

Finding an occurrence of a pattern (length m) takes $O(m \log n)$ time

Finding all occurrences takes $O(m \log n + \text{occ})$ time

where occ is the number of occurences

This can be further improved to $O(m + \log n + \text{occ})$ time

(using LCP queries which we will see in a future lecture)

We can construct the suffix array in $O(n)$ time