June 26, 2024
CPM 2024 @ Fukuoka, Japan

Contributed talk 4
[string algorithms and data structures]

Shortest cover after edit

Kazuki Mitani[1]

Graduated in March, ’24 🎓

Takuya Mieno[2]

Kazuhisa Seto[1]

Takashi Horiyama[1]

[1] Hokkaido University

[2] University of Electro-Communications

Basic definitions: borders and covers

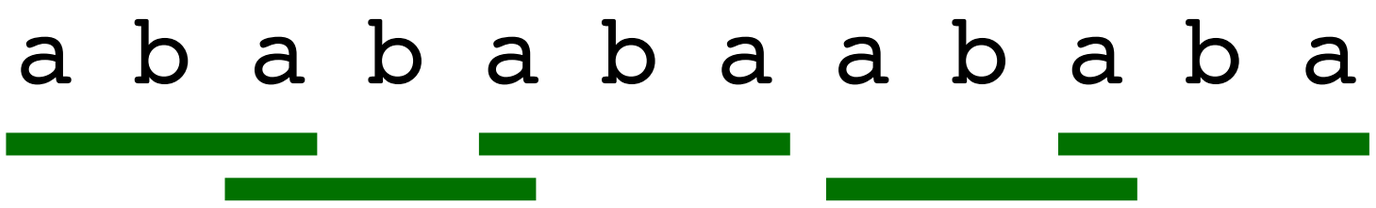
- A string B is called a ***border*** of another string T if B occurs both as a prefix and as a suffix of T.

$T_1 =$ a b a b a b a a b a b a

the border = the **longest** border

- A string C is called a ***cover*** (a.k.a. *quasi-period*) of T if each character of T lies within some occurrence of C.

$T_2 =$ a b a b a b a a b a b a



the cover = the **shortest** cover

- By definition, a cover of T must be a border of T.

Borders and covers when string is edited

- Observe how the borders and covers change as we **edit** T.

Borders:

$T_3 =$ a b a b a b a b a a a a b b b b a b a b a

$T'_3 =$ a b a b a b a b a a a a b a b a b a b a

Covers:

$T_4 =$ a b a b a a b a b a a b a b a a b a b a b a

$T'_4 =$ a b a b a a b a b a a a b a b a a

? Question ?

Can we quickly capture the changes in the **border**/the **cover**?

The problems and our results

Longest Border After-Edit (LBAE) query:

Preprocess : String T of length n.

Query : Substitute $T[i..j]$ with string w.

Output : The length of **the border** of the edited string T'.

Each query is applied to the **original** string T.

Shortest Cover After-Edit (SCAE) query:

Preprocess : String T of length n.

Query : Substitute $T[i..j]$ with string w.

Output : The length of **the cover** of the edited string T'.

Theorem (main result).

The longest border/shortest cover after-edit queries can be answered in $O(|w| \log n)$ time after $O(n)$-time preprocessing.

We claimed $O(|w| + \log n)$ time but there was a flaw…😨

Related work

- For **static** strings, the border and the cover can be computed in linear time [Knuth, Morris, Pratt, ’77], [Apostolico et al., ’91], [Breslauer, ’92].

- For **dynamic** strings:
  - The border:
    - $O(n^{o(1)})$ update/query time [Amir et al., ’19].
    - $O(\text{polylog}(n))$ update/query time (**w.h.p.**) is possible with *Internal Pattern Matching* and *Longest Common Extension* on a dynamic string [Charalampopoulos et al., ’20].
  - The cover:
    - $O(n)$ time for ***online*** string [Breslauer, ’92].
    - We couldn’t find any other previous work :(

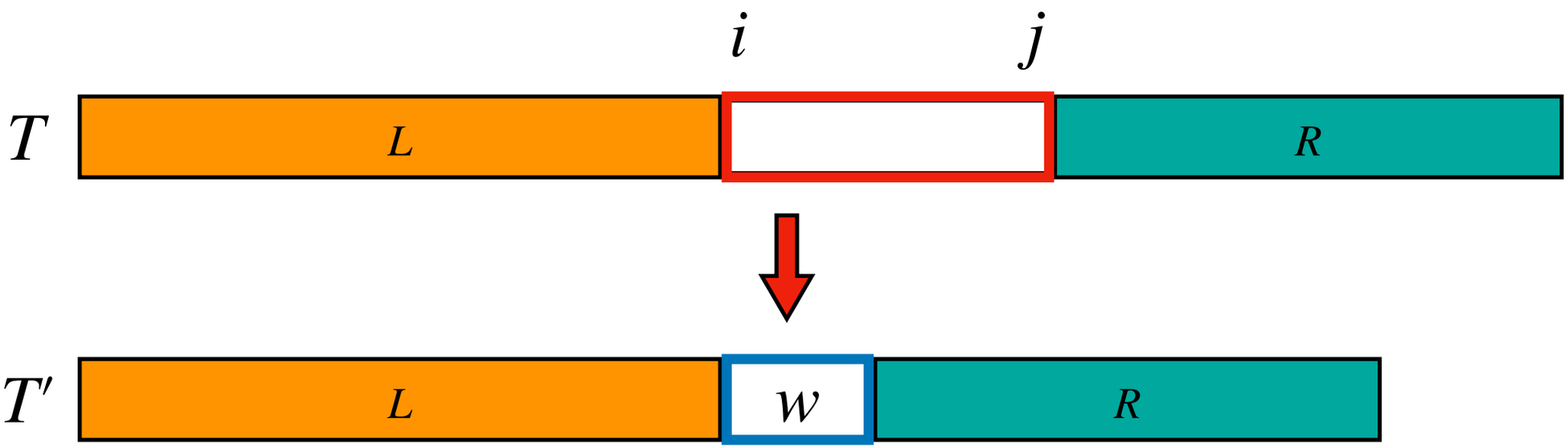
This talk

- We focus only on the SCAE queries.

Shortest Cover After-Edit (SCAE) query:

Preprocess :	String T of length n.
Query :	Substitute $T[i..j]$ with string w.
Output :	The length of **the cover** of the edited string T'.

- Let $T' = LwR$ where $L = T[1..i-1]$ and $R = T[j+1..n]$.
- Assume $|L| \geq |R|$ and $|w| \leq n/2$.
  - If $|w| > n/2$, a known $O(n) = O(|w|)$ time algorithm is optimal.



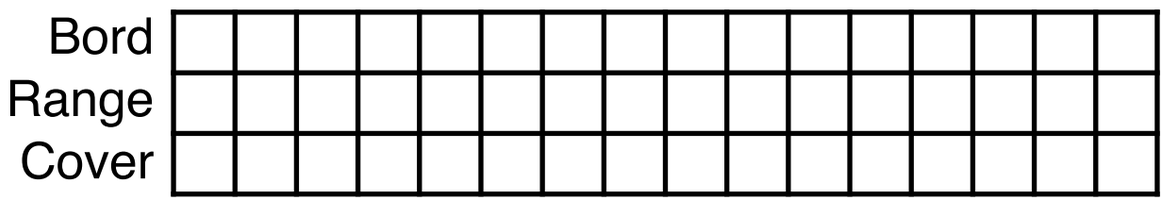
Overview of SCAE algorithm

The border of T'.

1. Compute $bord(T')$ by our LBAE algorithm (omitted in this talk);
2. Compute three arrays Bord, Range, and Cover for string Lw;
3. If $|bord(T')| > |T'|/2$:
4. Run subroutine for **periodic** case;
5. else:
6. Run subroutine for **non-periodic** case;

1.

2.



Three arrays: Bord, Range, Cover [Breslauer, '92]

- Breslauer's online algorithm maintains arrays Bord, Range, and Cover for an online text T defined as follows:
  - Bord$[i]$ stores the length of the border of $T[1..i]$.
  - Range$[i]$ stores the length of the longest prefix of T which can be covered by $T[1..i]$.
  - Cover$[i]$ stores the length of the cover of $T[1..i]$.

E.g.

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

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

Bord$[3] = |bord(\texttt{aba})| = 1$. Cover$[3] = |\texttt{aba}| = 3$.

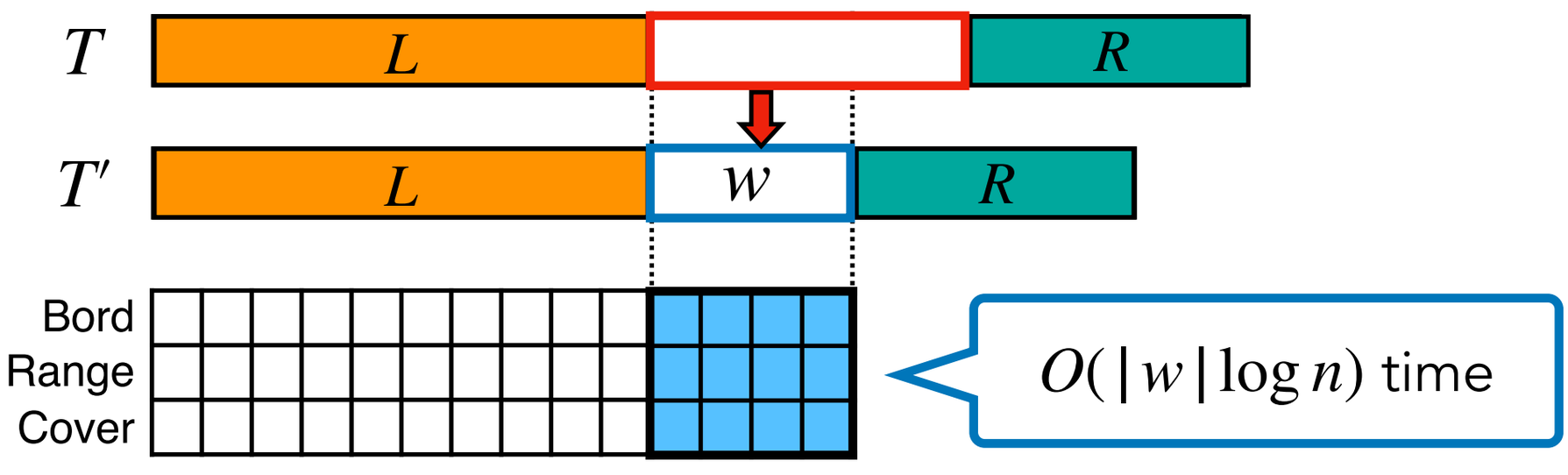
Range$[3] = 10$ (aba can cover $T[1..10]$ and cannot cover any longer prefix).

Updating three arrays for given edit operation

Bord[i]: the length of the border of $T[1..i]$.
Range[i]: the length of the longest prefix of T which can be covered by $T[1..i]$.
Cover[i]: the length of the cover of $T[1..i]$.

Lemma 1.

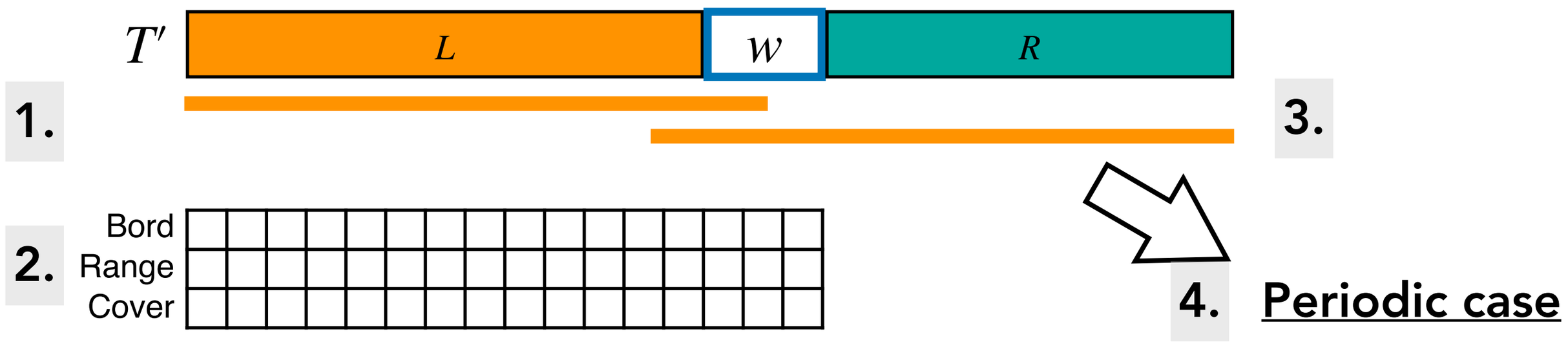
After $O(n)$-time preprocessing, given a substitution query, we can access any k elements of Bord/Range/Cover of Lw in a total of $O(|w| \log n + k)$ time.



Overview of SCAE algorithm

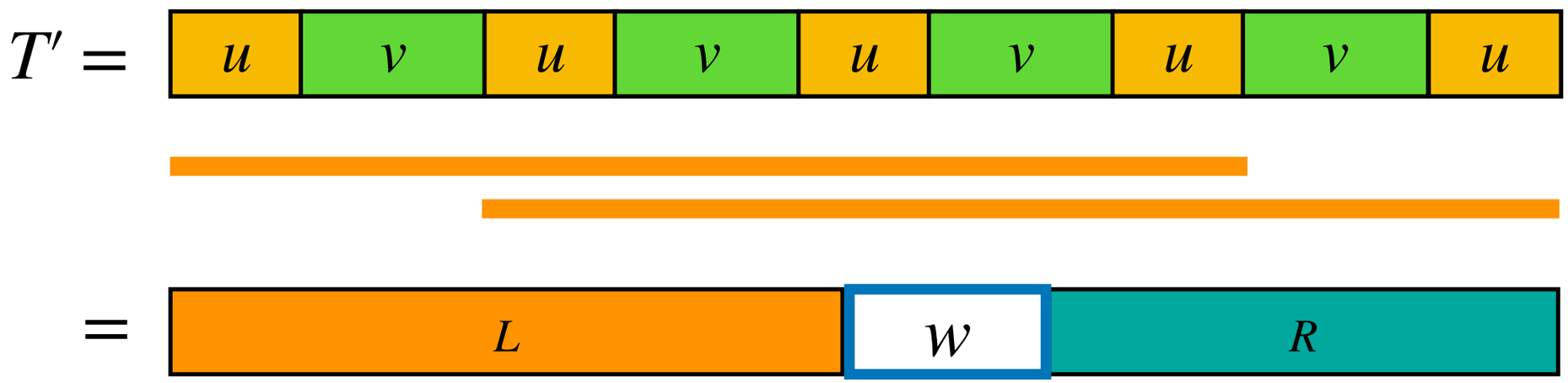
The border of T'.

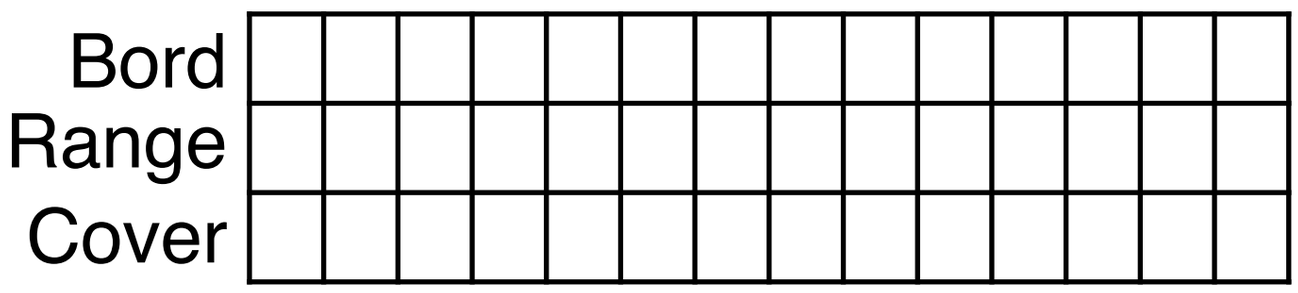
1. Compute $bord(T')$ by our LBAE algorithm (omitted in this talk);
2. Compute three arrays Bord, Range, and Cover for string Lw;
3. If $|bord(T')| > |T'|/2$:
4. Run subroutine for **periodic** case;
5. else:
6. Run subroutine for **non-periodic** case;



Periodic case: $|bord(T')| > |T'|/2$

- If $|bord(T')| > |T'|/2$ then $T' = (uv)^k u$ s.t. $bord(T') = (uv)^{k-1}u$.





Lemma 1

- In the periodic case, we need to have some discussions that take into account the ***periodicity*** of T' carefully…🤭
- Then, we can compute $cov(T')$ quickly by using **Range** array and **Cover** array of Lw.

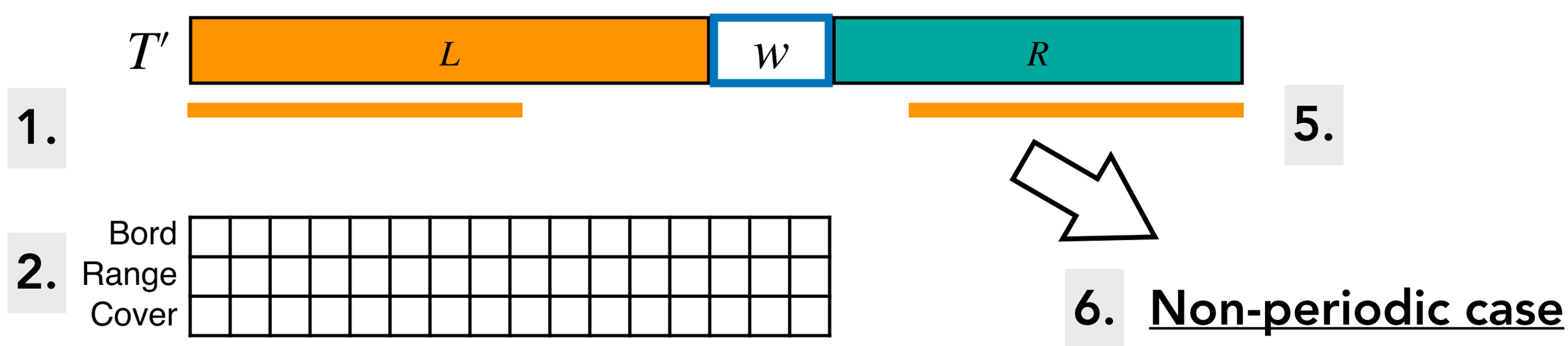
The cover of T'

Overview of SCAE algorithm

The border of T'.

1. Compute $bord(T')$ by our LBAE algorithm (omitted in this talk);
2. Compute three arrays Bord, Range, and Cover for string Lw;
3. If $|bord(T')| > |T'|/2$:
4. Run subroutine for **periodic** case;

5. else:
6. Run subroutine for **non-periodic** case;



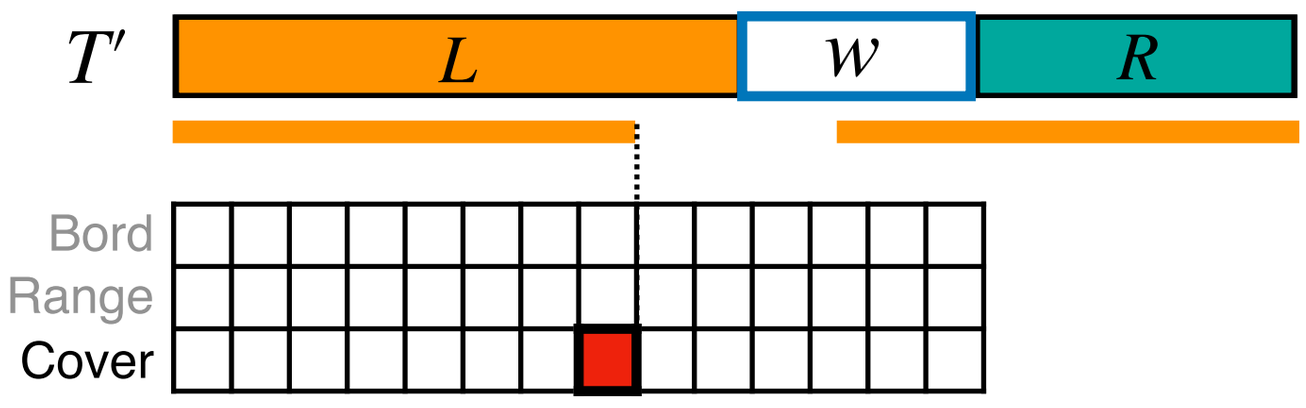
Non-periodic case: $|bord(T')| \leq |T'|/2$

- We use the following fact:

Lemma 2 ([Breslauer, '92]).

For any string T', $cov(T')$ is either $cov(bord(T'))$ or T' itself.

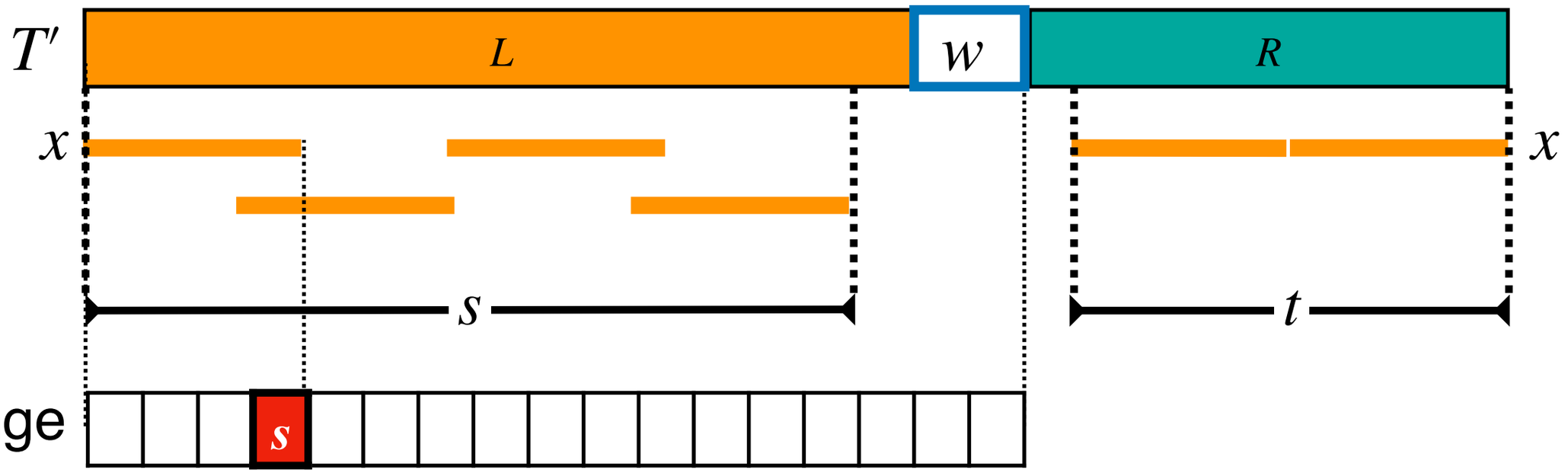
1. Compute $x = cov(bord(T'))$;
   - Since $|bord(T')| \leq |T'|/2 \leq |Lw|$, we can obtain $x = cov(bord(T'))$ by refering to **Cover**$[\,|bord(T')|\,]$.



2. Determine if $x = cov(bord(T'))$ covers T' or not;

Can $x = cov(bord(T'))$ cover T'? (1/3)

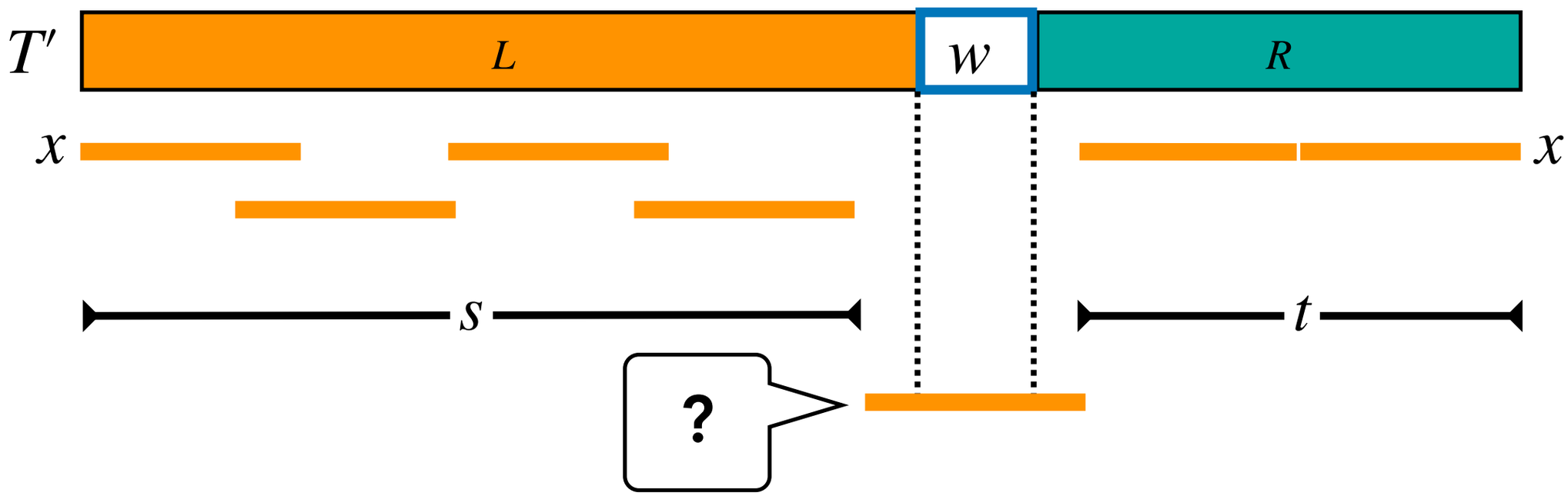
- Let s be the maximum length of the prefix of Lw that x can cover.
- Such s can be obtained by accessing Range$[|x|]$ of Lw.



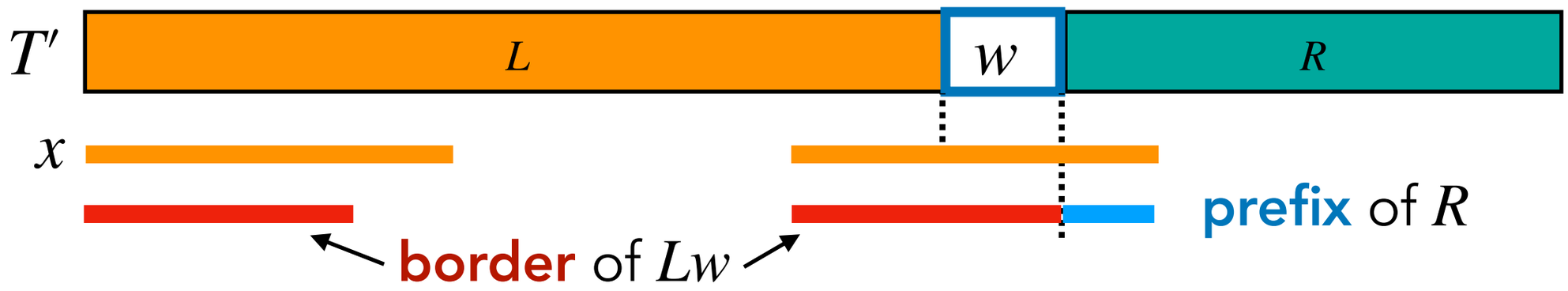
- Similarly, we can obtain the maximum length t of the suffix of wR that x can cover.
- If $s + t \geq |T'|$, then x is a cover of T'. Output x.
- Otherwise (like the figure above) … (👉 go to the next page.)

Can $x = cov(bord(T'))$ cover T'? (2/3)

- If $s + t < |T'|$, we check the existence of an occurrence of x beginning in L and ending in R.



- If such an occurrence exists, the occurrence is the concatenation of some **border** of Lw and some **prefix** of R as below.



Can $x = cov(bord(T'))$ cover T'? (3/3)

- Thus, we can check the existence of such an occurrence of x by applying an LCE query from **every border** of Lw **to the right**.

$O(|Lw|) = O(n)$ borders in the worst case…

- For speeding up, we utilize the ***periodicity*** of borders:

Fact

The set of borders of string T can be divided into $O(\log n)$ groups w.r.t. their smallest periods.

- For each group, extending at most **one** border is enough to find such x. Thus only $O(\log n)$ LCE queries are sufficient.

Overview of SCAE algorithm

1. Compute $bord(T')$ by our LBAE algorithm (omitted in this talk);
2. Compute three arrays Bord, Range, and Cover for string Lw;
3. If $|bord(T')| > |T'|/2$:
4. Run subroutine for **periodic** case;
5. else:
6. Run subroutine for **non-periodic** case;

$O(|w|\log n)$ time

$O(|w|\log n)$ time

$O(1)$ time

$O(\log n)$ time

Theorem (main result).

The longest border/shortest cover queries can be answered in $O(|w|\log n)$ time after $O(n)$-time preprocessing.

Conclusions and future work

Longest Border/Shortest Cover After-Edit (LBAE/SCAE) queries:

Preprocess : String T of length n.

Query : Substitute $T[i..j]$ with string w.

Output : The length of **the border/the cover** of the edited string T'.

Theorem (main result).

The longest border/shortest cover queries can be answered in $O(|w| \log n)$ time after $O(n)$-time preprocessing.

- Future work
  - Speeding up the query time.
    - More detailed analysis may lead $O(|w| + \log n)$ query time.
    - Can we further improve it to $O(|w| + \log \log n)$ time?
  - How can we compute the cover in **fully-dynamic setting**?