Online construction of the RLBWT
in time $O(n \lg(r))$ and compressed space $O(r \lg(n))$

[Policriti, Gigante & Prezza, LATA 2015]
[Ohno, Sakai, Takabatake, I & Sakamoto, J. Discrete Algorithms 52-53, 2018]
[Bannai, Gagie & I, CPM 2018]

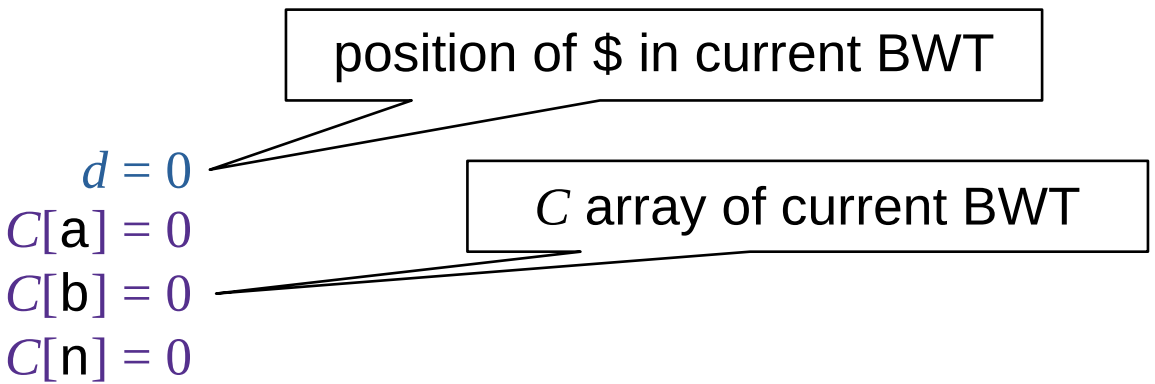
- **Goal**: Compute BWT of T=banana$
- **Approach**:

- **Goal**: Compute BWT of T=banana$
- **Approach**:
  - Begin with (trivial) BWT of $
  - From BWT of $$T[i+1, n]$, compute BWT of $$T[i, n]$ in time $O(\lg r)$

Text: banana**$**

0

BWT: **$**



Text: banan**a$**

0

BWT: **$**

1) Compute $LF(p) := rank_{\mathbf{a}}(d) + C[\mathbf{a}] = 0 + 0 = 0$

$d = 0$

$C[\text{a}] = 0$

$C[\text{b}] = 0$

$C[\text{n}] = 0$

Text: banan**a$**

0

BWT: **a**

1) Compute $LF(p) := rank_{\mathbf{a}}(d) + C[\mathbf{a}] = 0 + 0 = 0$
2) Replace **$** by **a**

$d = 0$
$C[\text{a}] = 0$
$C[\text{b}] = 0$
$C[\text{n}] = 0$

Text: banan**a$**

01

BWT: **a$**

1) Compute $LF(p) := rank_{\mathbf{a}}(d) + C[\mathbf{a}] = 0 + 0 = 0$
2) Replace **$** by **a** in current BWT
3) Re-insert **$** after $LF(p)$

$d = 0$
$C[\text{a}] = 0$
$C[\text{b}] = 0$
$C[\text{n}] = 0$

Text: banan**a$**

01

BWT: **a$**

1) Compute $LF(p) := rank_{\mathbf{a}}(d) + C[\mathbf{a}] = 0 + 0 = 0$
2) Replace **$** by **a** in current BWT
3) Re-insert **$** after $LF(p)$
4) Update d and C

$d = 1$
$C[\text{a}] = 0$
$C[\text{b}] = 1$
$C[\text{n}] = 1$

Text: banan$a

01

BWT: a$

$d = 1$

$C[\mathtt{a}] = 0$

$C[\mathtt{b}] = 1$

$C[\mathtt{n}] = 1$

Text: banan$a

01

BWT: a$

1) Compute $LF(p) := rank_{n}(d) + C[n] = 0 + 1 = 1$

$d = 1$

$C[a] = 0$

$C[b] = 1$

$C[n] = 1$

Text: banan$a

012

BWT: an$

1) Compute $LF(p) := rank_{\mathbf{n}}(d) + C[\mathbf{n}] = 0 + 1 = 1$
2) Replace **$** by **n** in current BWT
3) Re-insert **$** after $LF(p)$
4) Update d and C

$d = 2$
$C[\text{a}] = 0$
$C[\text{b}] = 1$
$C[\text{n}] = 1$

Text: bana**$**na

012

BWT: an**$**

$d = 2$

C[a] = 0

C[b] = 1

C[n] = 1

Text: ban**a**$na

012

BWT: an$

1) Compute $LF(p) := rank_{\mathbf{a}}(d) + C[\mathbf{a}] = 1 + 0 = 1$

$d = 2$

$C[\text{a}] = 0$

$C[\text{b}] = 1$

$C[\text{n}] = 1$

Text: ban**a**$na

0123

BWT: an$**a**

1) Compute $LF(p) := rank_{\mathbf{a}}(d) + C[\mathbf{a}] = 1 + 0 = 0$
2) Replace **$** by **a** in current BWT
3) Re-insert **$** after $LF(p)$
4) Update d and C

$d = 2$
$C[\text{a}] = 0$
$C[\text{b}] = 2$
$C[\text{n}] = 2$

Text: ban**$**ana

0123

BWT: an**$**a

$d = 2$

$C[\mathrm{a}] = 0$

$C[\mathrm{b}] = 2$

$C[\mathrm{n}] = 2$

Text: ban$ana

0123

BWT: an$a

1) Compute $LF(p) := rank_{n}(d) + C[\mathbf{n}] = 1 + 2 = 3$

$d = 2$

$C[a] = 0$

$C[b] = 2$

$C[n] = 2$

Text: ban$ana

01234

BWT: anna$

1) Compute $LF(p) := rank_{\mathbf{n}}(d) + C[\mathbf{n}] = 1 + 2 = 3$
2) Replace **$** by **n** in current BWT
3) Re-insert **$** after $LF(p)$
4) Update d and C

$d = 4$
$C[\text{a}] = 0$
$C[\text{b}] = 2$
$C[\text{n}] = 2$

Text: ba$nana

01234

BWT: anna$

$d = 4$

$C[\mathrm{a}] = 0$

$C[\mathrm{b}] = 2$

$C[\mathrm{n}] = 2$

Text: **b a $ n a n a**

01234

BWT: **a n n a $**

1) Compute $LF(p) := rank_{\mathbf{a}}(d) + C[\mathbf{a}] = 2 + 0 = 2$

$d = 4$

$C[\text{a}] = 0$

$C[\text{b}] = 2$

$C[\text{n}] = 2$

Text: b**a**$nana

012345

BWT: ann$a**a**

1) Compute $LF(p) := rank_{\mathbf{a}}(d) + C[\mathbf{a}] = 2 + 0 = 2$
2) Replace **$** by **a** in current BWT
3) Re-insert **$** after $LF(p)$
4) Update d and C

$d = 3$
$C[\text{a}] = 0$
$C[\text{b}] = 3$
$C[\text{n}] = 3$

Text: b$anana

012345

BWT: ann$aa

$d = 3$

$C[a] = 0$

$C[b] = 3$

$C[n] = 3$

Text: **b$**anana

012345

BWT: ann**$**aa

1) Compute $LF(p) := rank_{\mathbf{b}}(d) + C[\mathbf{b}] = 0 + 3 = 3$

$d = 3$

$C[\text{a}] = 0$

$C[\text{b}] = 3$

$C[\text{n}] = 3$

Text: **b$**anana

0123456

BWT: ann**b$**aa

1) Compute $LF(p) := rank_{\mathbf{b}}(d) + C[\mathbf{b}] = 0 + 3 = 3$
2) Replace **$** by **b** in current BWT
3) Re-insert **$** after $LF(p)$
4) Update d and C

$d = 4$
$C[\text{a}] = 0$
$C[\text{b}] = 3$
$C[\text{n}] = 4$

Text: $banana

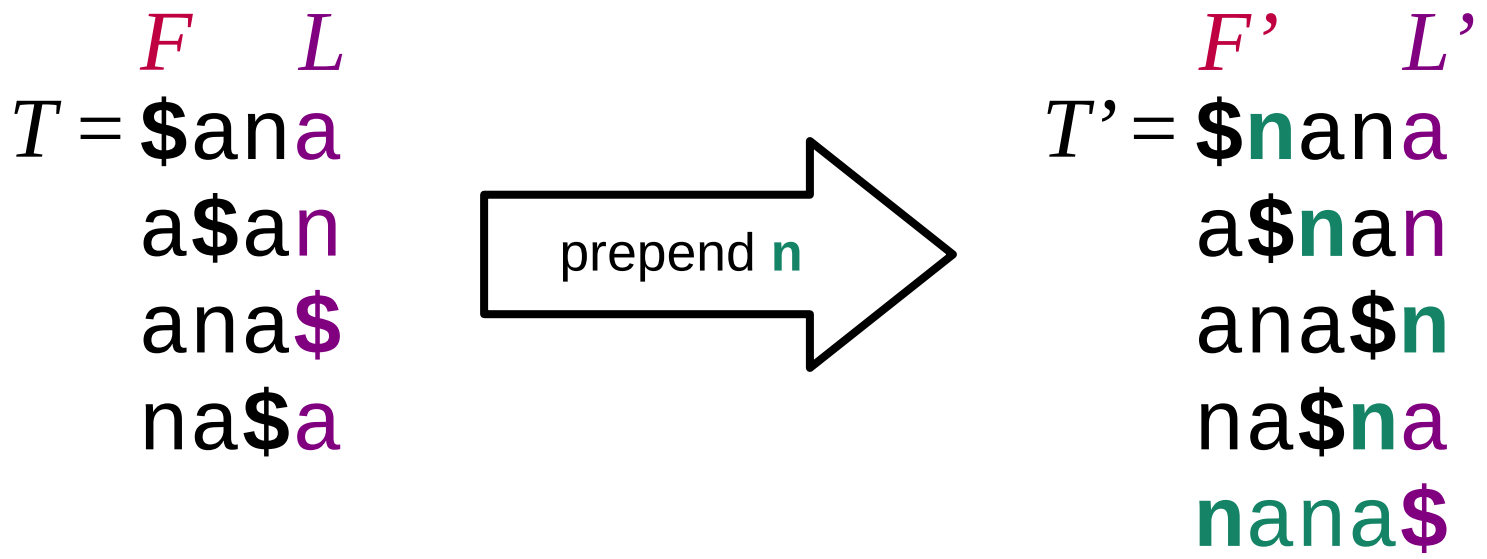
0123456

BWT: annb$aa

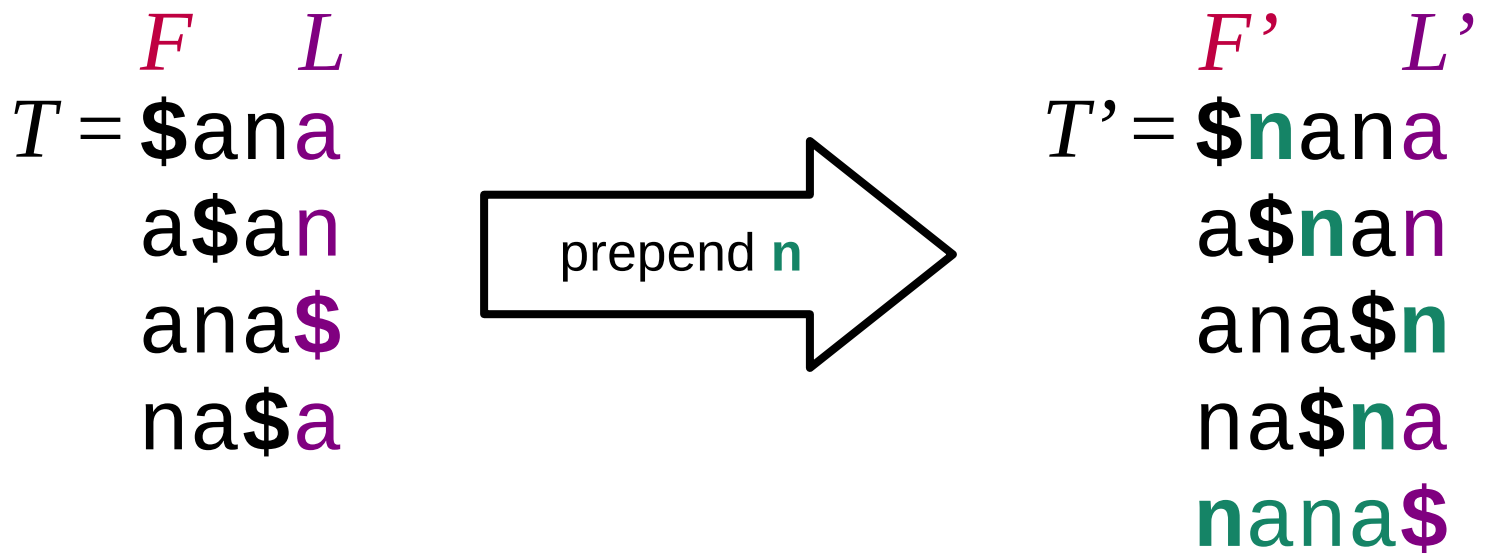
Why does it work?

$$T = \begin{matrix} F & L \\ \$\text{an} & \text{a} \\ \text{a}\$\text{a} & \text{n} \\ \text{ana}\$ & \$ \\ \text{na}\$\text{a} & \text{a} \end{matrix}$$

Why does it work?

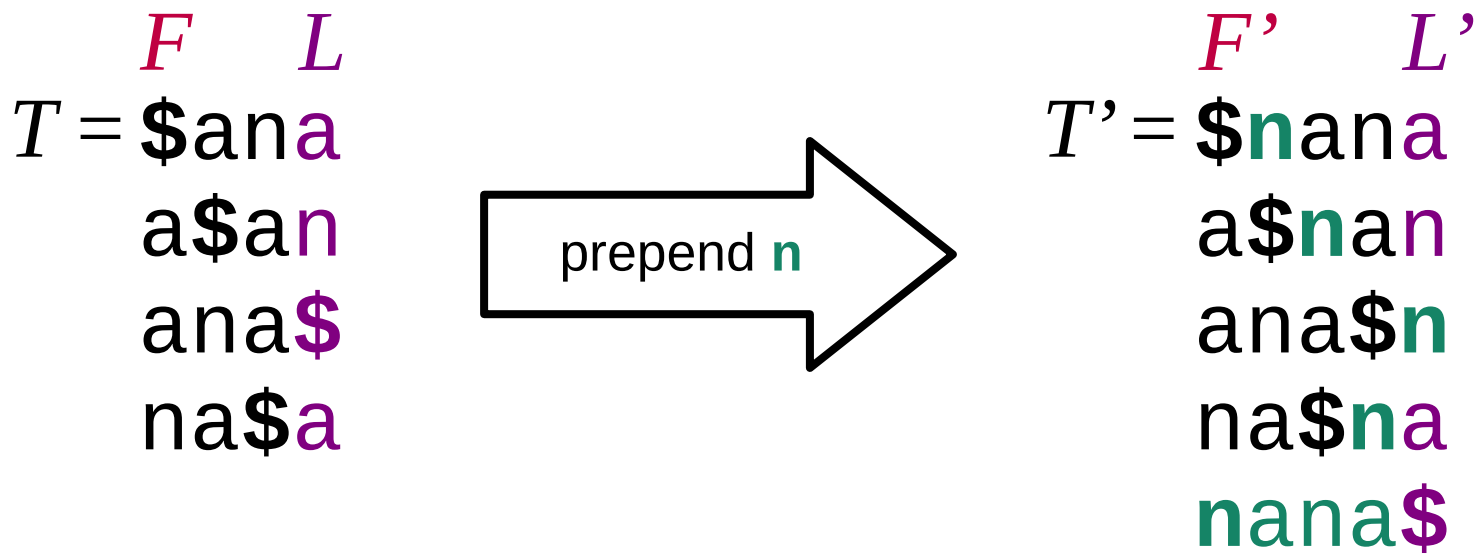


Why does it work?



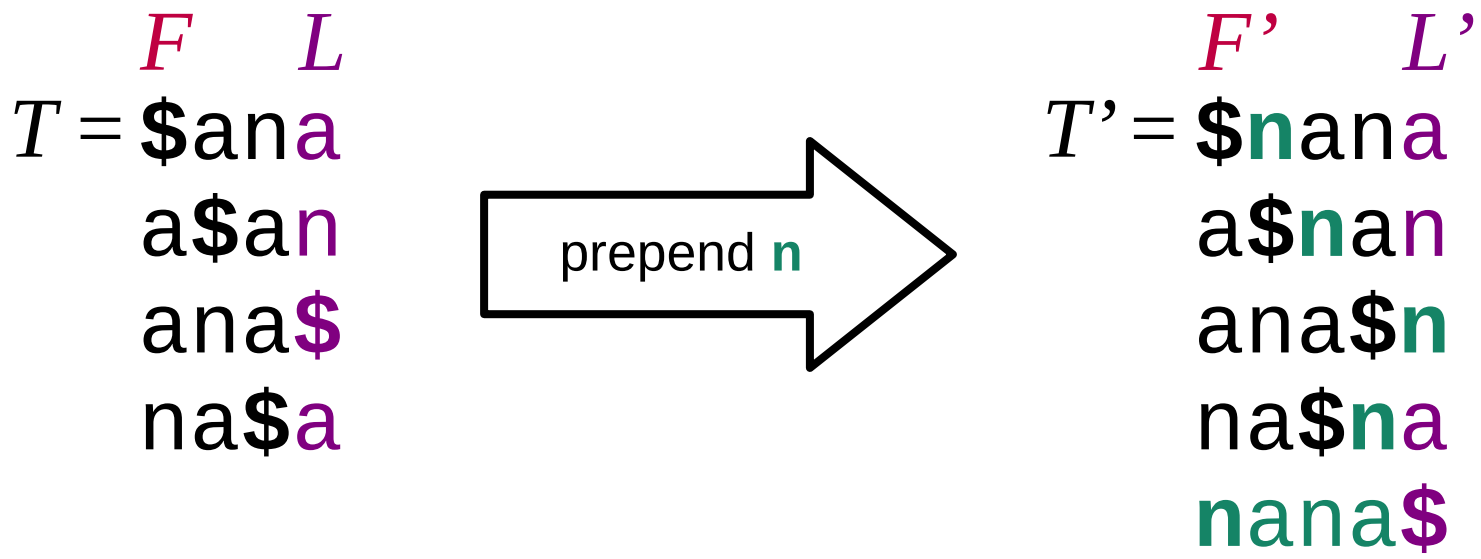
- Lex. order of $\$T$'s rotations is retained thanks to **$**

Why does it work?



- Lex. order of **$***T*'s rotations is retained thanks to **$**
- Lex. rank of new rotation *T*'**$** is one greater than of the rotation of *T* starting with the last occurence of the new character before **$**

Why does it work?



- Lex. order of **$***T*'s rotations is retained thanks to **$**
- Lex. rank of new rotation *T*'**$** is one greater than of the rotation of *T* starting with the last occurence of the new character before **$**
- Can be located using $LF(p)$

Time & Space

L = annb$aa

- n=7, r=5

Time & Space

L = annb$aa

- n=7, r=5

H := anb$a

- $r\ lg(\sigma)$ = ***O(r lg(n))*** bits

Time & Space

L = annb$aa

- n=7, r=5

H := anb$a

- $r\ lg(\sigma)$ = ***O(r lg(n))*** bits

G := 1101110

- n bits

$G_{\$}$:= 1

- ***O(r lg(n))*** bits with gap encoding

G_{a} := 110

G_{b} := 1

G_{n} := 10

Time & Space

L = annb$aa

- n=7, r=5

H := anb$a

- $r\ lg(\sigma)$ = ***O(r lg(n))*** bits

G := 1101110

- n bits

$G_{\$}$:= 1

G_{a} := 110

G_{b} := 1

G_{n} := 10

- ***O(r lg(n))*** bits with gap encoding

- ***O(lg r)*** access & update time, e.g., using balanced trees