# Fully-Compressed Suffix Trees

Luís M. S. Russo Gonzalo Navarro Arlindo L. Oliveira

INESC-ID/IST
{**lsr,aml**}**@algos.inesc-id.pt**

Dept. of Computer Science, University of Chile
**gnavarro@dcc.uchile.cl**

Workshop on Compression, Text, and Algorithms 2007

# Outline

1. Motivation
   - The Problem We Studied
   - Previous Work
2. Our Contribution
   - Performance
   - The kernel Operations
   - Further Operations
3. Conclusions
   - Summary

# Suffix Trees are Important



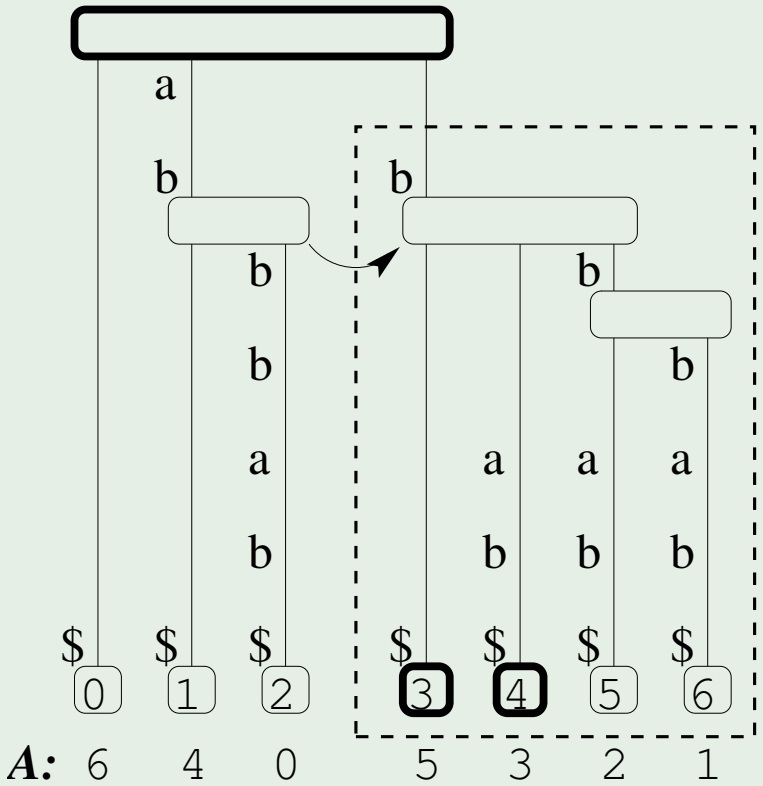
Suffix trees are important for several string problems:

- pattern matching
- longest common substring
- super maximal repeats
- bioinformatics applications
- etc

# Suffix Trees are Important

23 min

## Example (Suffix Tree for *abbbab*)



# Representation Problems



**Problem (Suffix Trees need too much space)**

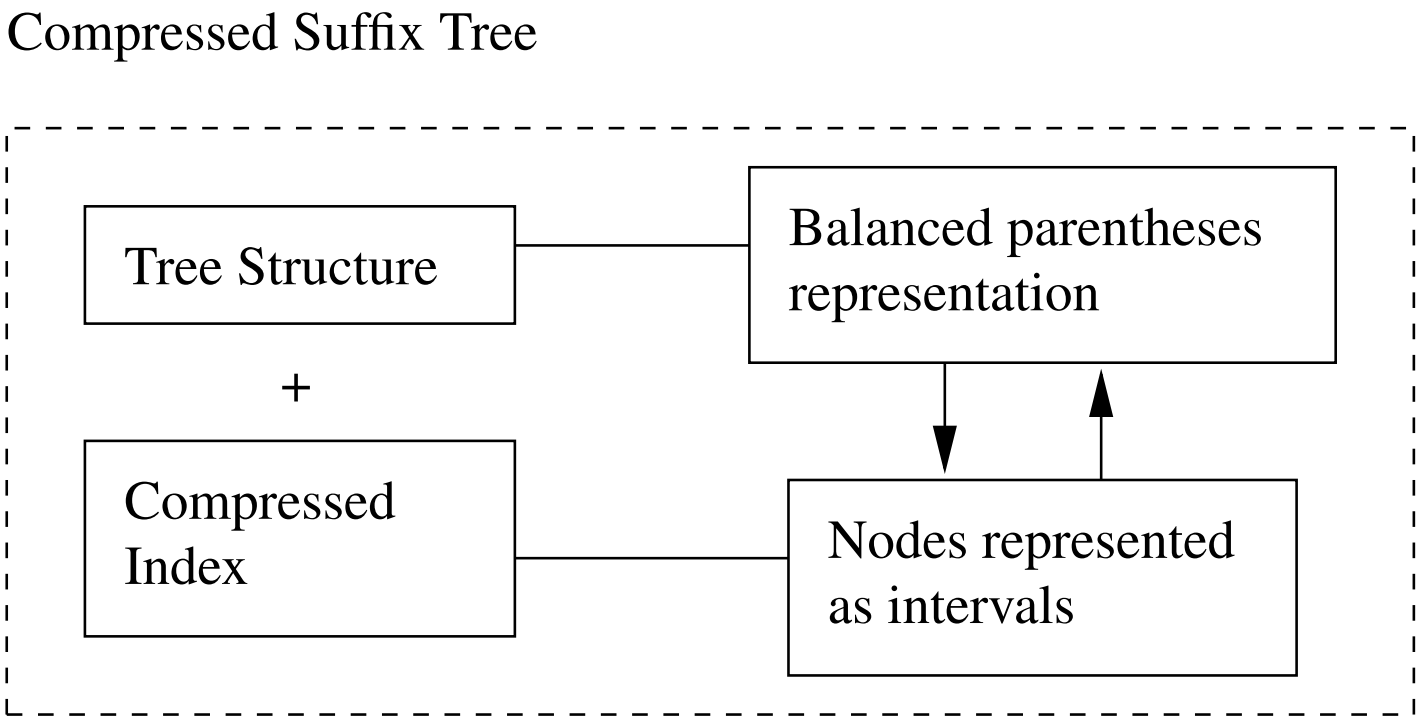
*Pointer based representations require $O(n \log n)$ bits.*

This is much larger than the indexed string.
State of the art implementations require $[8, 10]n \log \sigma$ bits.

# Compressed Representations

21 min

Sadakane proposed a new way to represent suffix trees.



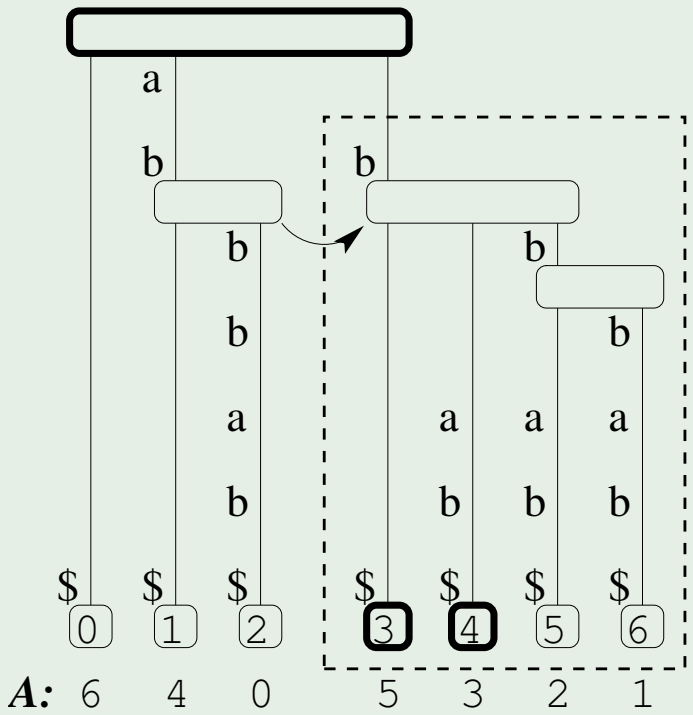
# Node Representation

21 min

A node represented as an interval of leaves of a suffix tree.

## Example

Interval [3, 6] represents node *b*.



# Compressed Indexes



Compressed indexes are compressed representations of the leaves of a suffix tree.
Their success relies on:

- `Succinct structures`, based on RANK and SELECT.
- `Data compression`, that represent $T$ in $O(uH_k)$ bits.

**Examples**

FM-index, Compressed Suffix Arrays, LZ-index, etc.

Sadakane used compressed suffix arrays.
We need a compressed index that supports $\psi$ and LF.
For example the Alphabet-Friendly FM-Index.

# Overall Performance



$\sigma = O(\text{polylog}(n))$

| | Sadakane's | Ours |
|---|---|---|
| Space in bits | $nH_k + 6n + o(n \log \sigma)$ | $nH_k + o(n \log \sigma)$ |
| SDEP/LOCATE | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| COUNT/ANCESTOR | 1 | 1 |
| PARENT/FCHILD/ | 1 | $(\log_\sigma \log n) \log n$ |
| SLINK | 1 | $(\log_\sigma \log n) \log n$ |
| SLINK$^i$ | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| LETTER$(v, i)$ | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| LCA | 1 | $(\log_\sigma \log n) \log n$ |
| CHILD | $(\log \log n) \log n$ | $(\log \log n)^2 \log_\sigma n$ |
| TDEP | 1 | $((\log_\sigma \log n) \log n)^2$ |
| LAQT | 1 | $((\log_\sigma \log n) \log n)^2$ |
| LAQS | — | $(\log_\sigma \log n) \log n$ |
| WEINERLINK | 1 | 1 |

# Overall Performance



$\sigma = O(\text{polylog}(n))$

| | Sadakane's | Ours |
|---|---|---|
| Space in bits | $nH_k + 6n + o(n \log \sigma)$ | $nH_k + o(n \log \sigma)$ |
| SDEP/LOCATE | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| COUNT/ANCESTOR | 1 | 1 |
| PARENT/FCHILD/ | 1 | $(\log_\sigma \log n) \log n$ |
| SLINK | 1 | $(\log_\sigma \log n) \log n$ |
| SLINK$^i$ | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| LETTER$(v, i)$ | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| LCA | 1 | $(\log_\sigma \log n) \log n$ |
| CHILD | $(\log \log n) \log n$ | $(\log \log n)^2 \log_\sigma n$ |
| TDEP | 1 | $((\log_\sigma \log n) \log n)^2$ |
| LAQT | 1 | $((\log_\sigma \log n) \log n)^2$ |
| LAQS | — | $(\log_\sigma \log n) \log n$ |
| WEINERLINK | 1 | 1 |

# Overall Performance



$\sigma = O(\text{polylog}(n))$

| | Sadakane's | Ours |
|---|---|---|
| Space in bits | $nH_k + 6n + o(n \log \sigma)$ | $nH_k + o(n \log \sigma)$ |
| SDEP/LOCATE | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| COUNT/ANCESTOR | 1 | 1 |
| PARENT/FCHILD/ | 1 | $(\log_\sigma \log n) \log n$ |
| SLINK | 1 | $(\log_\sigma \log n) \log n$ |
| SLINK$^i$ | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| LETTER$(v, i)$ | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| LCA | 1 | $(\log_\sigma \log n) \log n$ |
| CHILD | $(\log \log n) \log n$ | $(\log \log n)^2 \log_\sigma n$ |
| TDEP | 1 | $((\log_\sigma \log n) \log n)^2$ |
| LAQT | 1 | $((\log_\sigma \log n) \log n)^2$ |
| LAQS | — | $(\log_\sigma \log n) \log n$ |
| WEINERLINK | 1 | 1 |

# Overall Performance

18 min

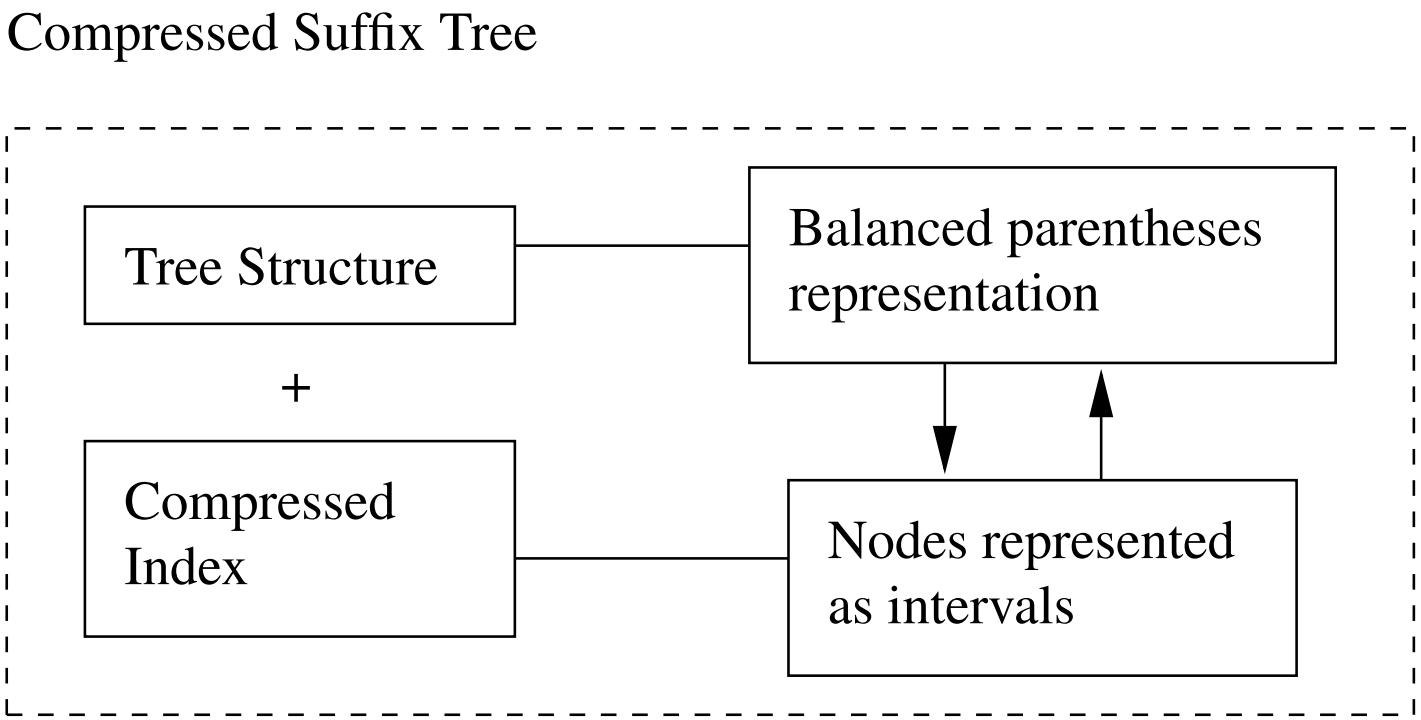
$\sigma = O(\text{polylog}(n))$

| | Sadakane's | Ours |
|---|---|---|
| Space in bits | $nH_k + 6n + o(n \log \sigma)$ | $nH_k + o(n \log \sigma)$ |
| SDEP/LOCATE | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| COUNT/ANCESTOR | 1 | 1 |
| PARENT/FCHILD/ | 1 | $(\log_\sigma \log n) \log n$ |
| SLINK | 1 | $(\log_\sigma \log n) \log n$ |
| SLINK$^i$ | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| LETTER$(v, i)$ | $(\log_\sigma \log n) \log n$ | $(\log_\sigma \log n) \log n$ |
| LCA | 1 | $(\log_\sigma \log n) \log n$ |
| CHILD | $(\log \log n) \log n$ | $(\log \log n)^2 \log_\sigma n$ |
| TDEP | 1 | $((\log_\sigma \log n) \log n)^2$ |
| LAQT | 1 | $((\log_\sigma \log n) \log n)^2$ |
| LAQS | — | $(\log_\sigma \log n) \log n$ |
| WEINERLINK | 1 | 1 |

# Sampling



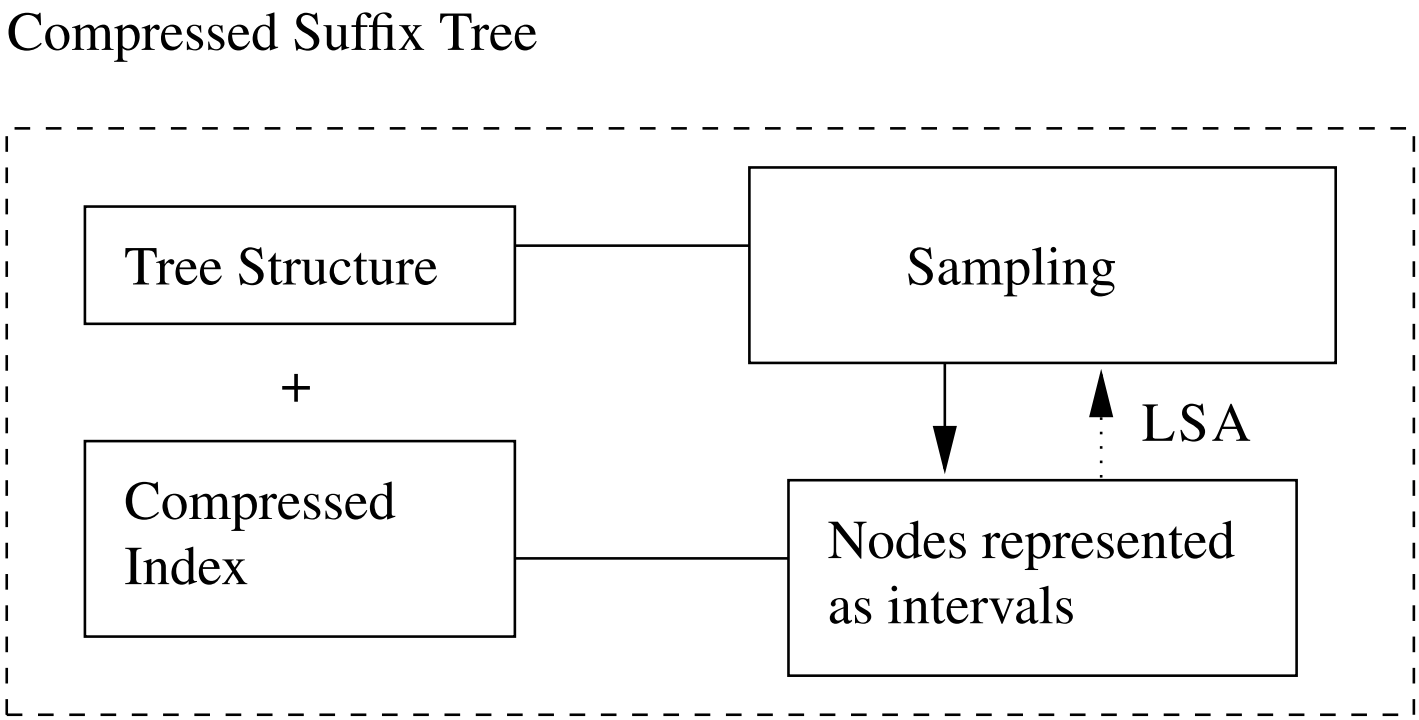
We use sampling instead of balanced parentheses.



# Sampling



We use sampling instead of balanced parentheses.



# Sampling



The sampling has the property that in any sequence

- $v$
- $\textsc{SLink}(v)$
- $\textsc{SLink}(\textsc{SLink}(v))$
- $\textsc{SLink}(\textsc{SLink}(\textsc{SLink}(v)))$
- ...

of size $\delta$ there is at least one sampled node.

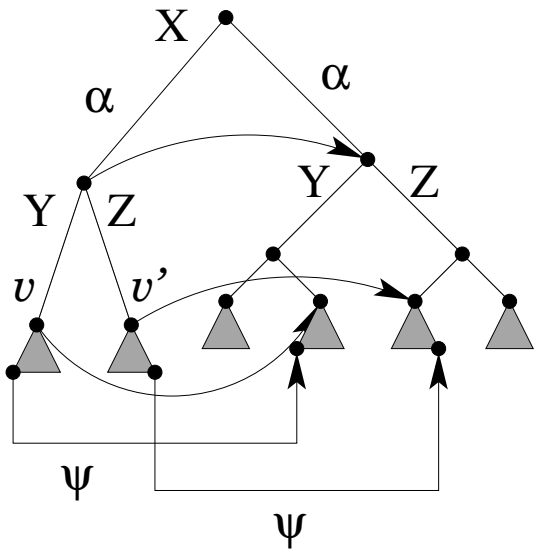
# LCA and SLINK



## Lemma

*When* $\mathrm{LCA}(v, v') \neq \mathrm{ROOT}$ *we have that:*

$$\mathrm{SLINK}(\mathrm{LCA}(v, v')) = \mathrm{LCA}(\mathrm{SLINK}(v), \mathrm{SLINK}(v'))$$



# Fundamental lemma



## Lemma

*If* $\text{SLINK}^r(\text{LCA}(v, v')) = \text{ROOT}$, *and let* $d = \min(\delta, r + 1)$.
*Then* $\text{SDEP}(\text{LCA}(v, v')) =$

$$\max_{0 \le i < d}\{i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))\}$$

## Proof.

$$\begin{aligned}
&\text{SDEP}(\text{LCA}(v, v')) \\
&\quad = i + \text{SDEP}(\text{SLINK}^i(\text{LCA}(v, v'))) \\
&\quad = i + \text{SDEP}(\text{LCA}(\text{SLINK}^i(v), \text{SLINK}^i(v'))) \\
&\quad \ge i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))
\end{aligned}$$

The last inequality is an equality for some $i \le d$. □

# Fundamental lemma

12 min

## Lemma

*If* $\text{SLINK}^r(\text{LCA}(v, v')) = \text{ROOT}$, *and let* $d = \min(\delta, r + 1)$.
*Then* $\text{SDEP}(\text{LCA}(v, v')) =$

$$\max_{0 \le i < d}\{i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))\}$$

## Proof.

$$\begin{aligned}
&\text{SDEP}(\text{LCA}(v, v')) \\
&\quad = i + \text{SDEP}(\text{SLINK}^i(\text{LCA}(v, v'))) \\
&\quad = i + \text{SDEP}(\text{LCA}(\text{SLINK}^i(v), \text{SLINK}^i(v'))) \\
&\quad \ge i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))
\end{aligned}$$

The last inequality is an equality for some $i \le d$. □

# Fundamental lemma



## Lemma

*If* $\text{SLINK}^r(\text{LCA}(v, v')) = \text{ROOT}$, *and let* $d = \min(\delta, r + 1)$.
*Then* $\text{SDEP}(\text{LCA}(v, v'))?$

$$\max_{0 \le i < d}\{i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))\}$$

## Proof.

$$\begin{aligned}
&\text{SDEP}(\text{LCA}(v, v')) \\
&\quad = i + \text{SDEP}(\text{SLINK}^i(\text{LCA}(v, v'))) \\
&\quad = i + \text{SDEP}(\text{LCA}(\text{SLINK}^i(v), \text{SLINK}^i(v'))) \\
&\quad \ge i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))
\end{aligned}$$

The last inequality is an equality for some $i \le d$. □

# Fundamental lemma



## Lemma

*If* $\text{SLINK}^r(\text{LCA}(v, v')) = \text{ROOT}$, *and let* $d = \min(\delta, r + 1)$.
*Then* $\text{SDEP}(\text{LCA}(v, v'))?$

$$\max_{0 \le i < d}\{i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))\}$$

## Proof.

$$\begin{aligned}
&\text{SDEP}(\text{LCA}(v, v')) \\
&\quad = i + \text{SDEP}(\text{SLINK}^i(\text{LCA}(v, v'))) \\
&\quad = i + \text{SDEP}(\text{LCA}(\text{SLINK}^i(v), \text{SLINK}^i(v'))) \\
&\quad \ge i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))
\end{aligned}$$

The last inequality is an equality for some $i \le d$. □

# Fundamental lemma



## Lemma

*If* $\text{SLINK}^r(\text{LCA}(v, v')) = \text{ROOT}$, *and let* $d = \min(\delta, r + 1)$.
*Then* $\text{SDEP}(\text{LCA}(v, v')) \geq$
$\max_{0 \leq i < d}\{i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))\}$

## Proof.

$$\begin{aligned}
&\text{SDEP}(\text{LCA}(v, v')) \\
&\quad = i + \text{SDEP}(\text{SLINK}^i(\text{LCA}(v, v'))) \\
&\quad = i + \text{SDEP}(\text{LCA}(\text{SLINK}^i(v), \text{SLINK}^i(v'))) \\
&\quad \geq i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))
\end{aligned}$$

The last inequality is an equality for some $i \leq d$. □

# Fundamental lemma

12 min

## Lemma

*If* $\text{SLINK}^r(\text{LCA}(v, v')) = \text{ROOT}$, *and let* $d = \min(\delta, r + 1)$.
*Then* $\text{SDEP}(\text{LCA}(v, v')) =$

$$\max_{0 \le i < d}\{i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))\}$$

## Proof.

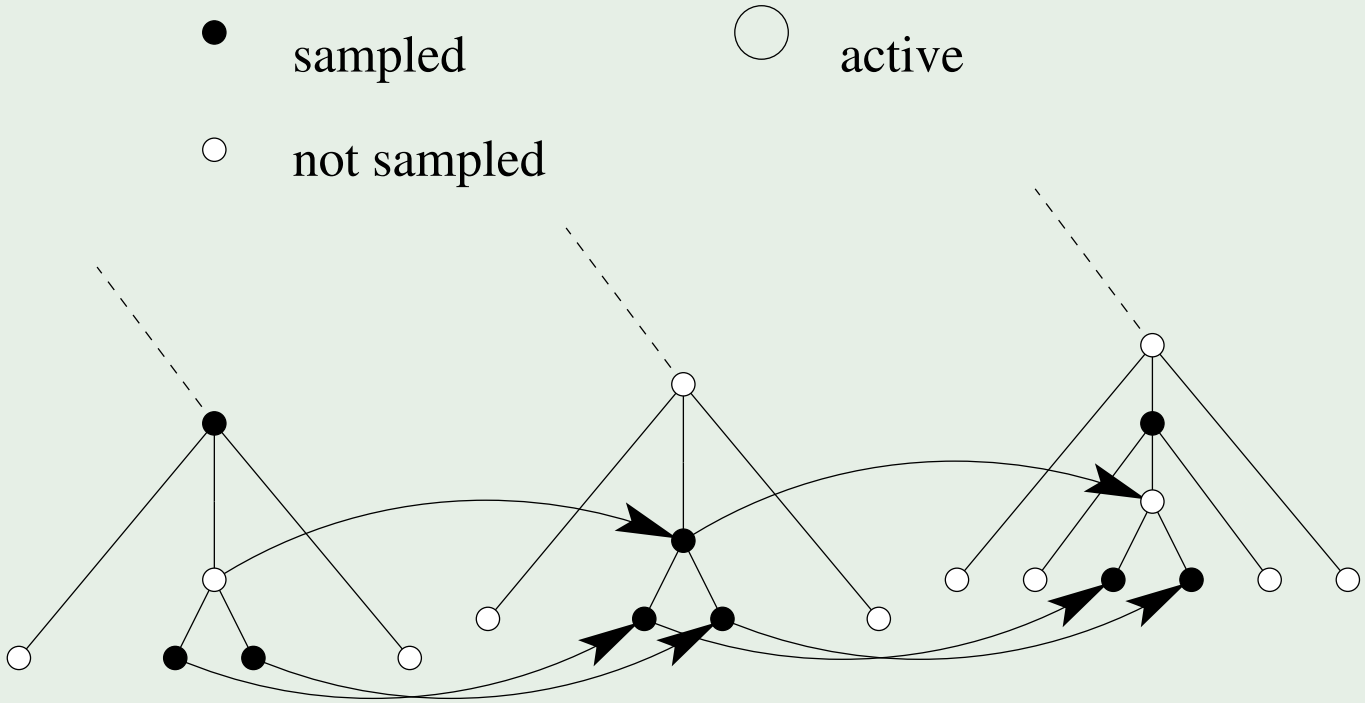
$$\begin{aligned}
&\text{SDEP}(\text{LCA}(v, v')) \\
&\quad = i + \text{SDEP}(\text{SLINK}^i(\text{LCA}(v, v'))) \\
&\quad = i + \text{SDEP}(\text{LCA}(\text{SLINK}^i(v), \text{SLINK}^i(v'))) \\
&\quad \ge i + \text{SDEP}(\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))
\end{aligned}$$

The last inequality is an equality for some $i \le d$. □

# Fundamental lemma

11 min

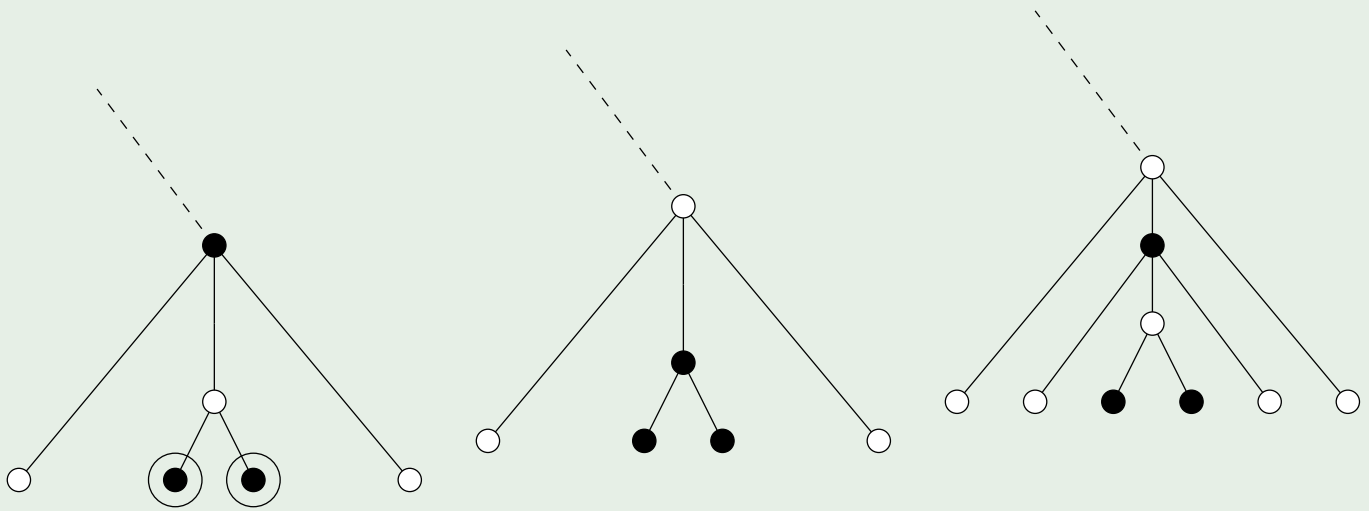
## Example ($\delta = 3$)



# Fundamental lemma

11 min

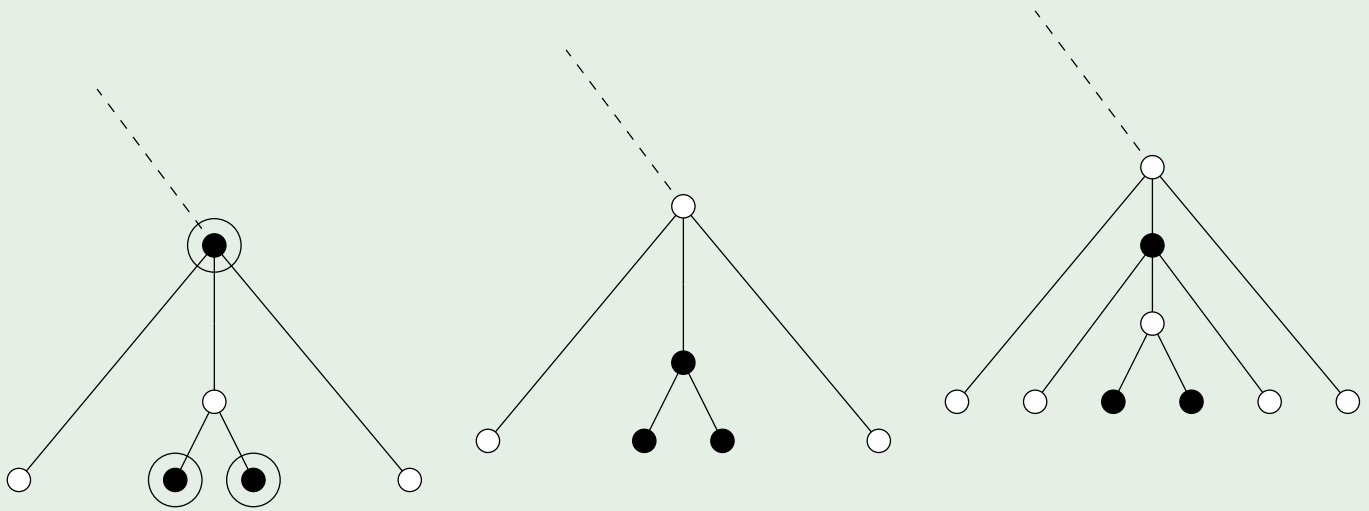
## Example ($\delta = 3$)



# Fundamental lemma

11 min

## Example ($\delta = 3$)

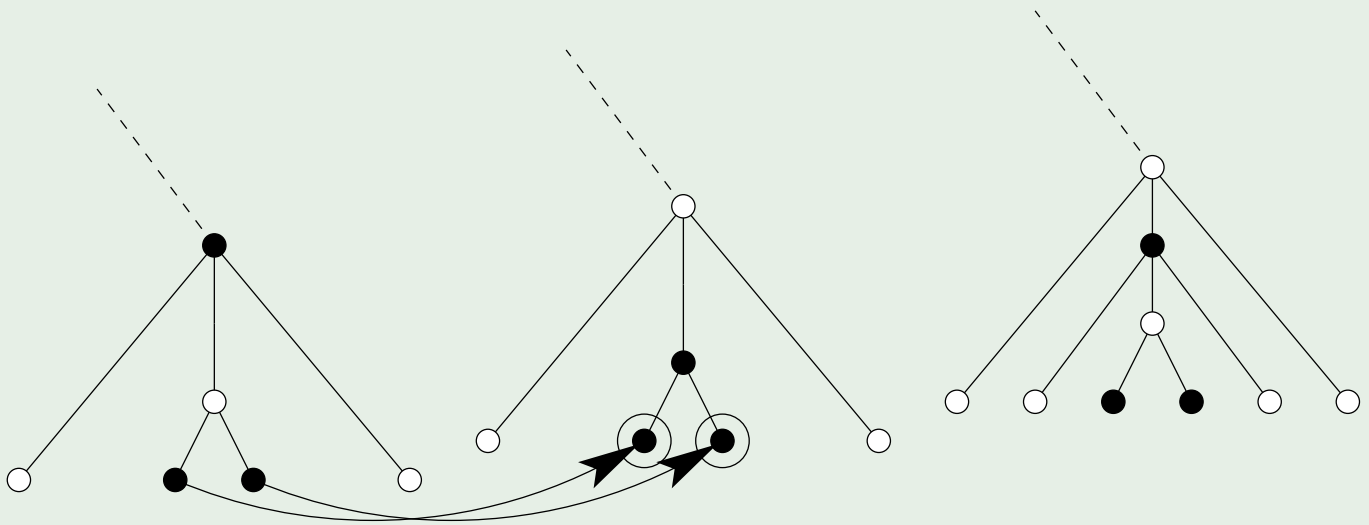


SDEP : 5

# Fundamental lemma

11 min

## Example ($\delta = 3$)

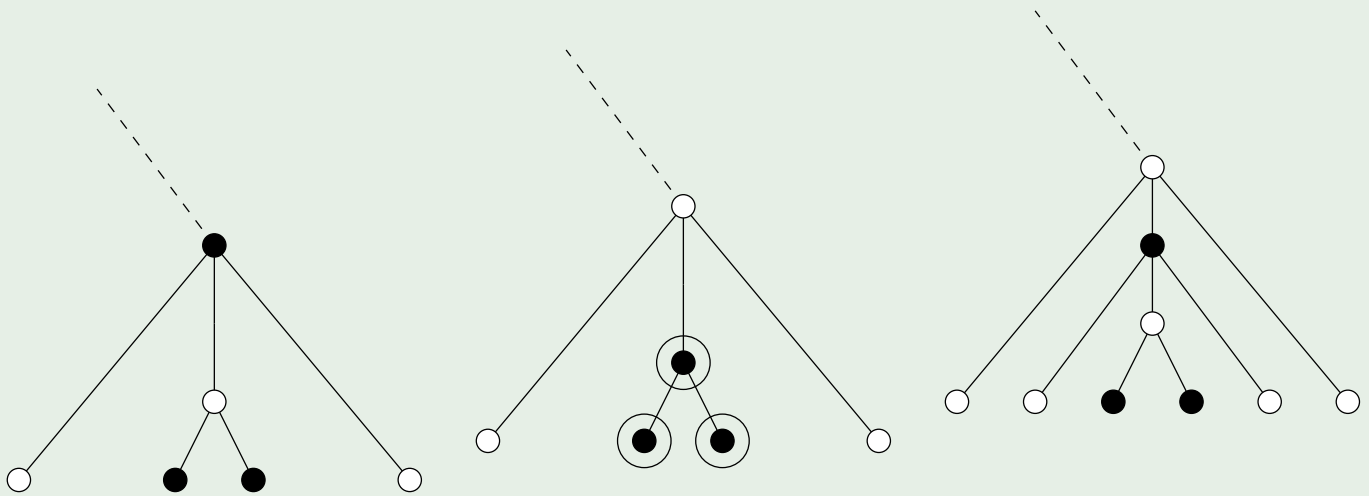


SDEP : 5

# Fundamental lemma

11 min

## Example ($\delta = 3$)

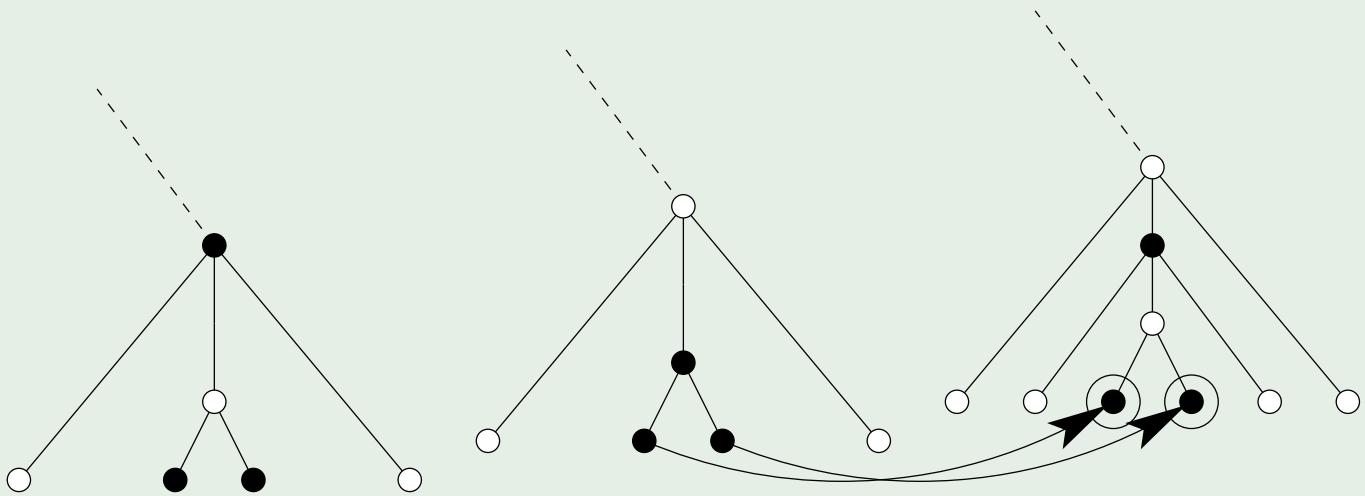


SDEP : 5 10

# Fundamental lemma

11 min

## Example ($\delta = 3$)

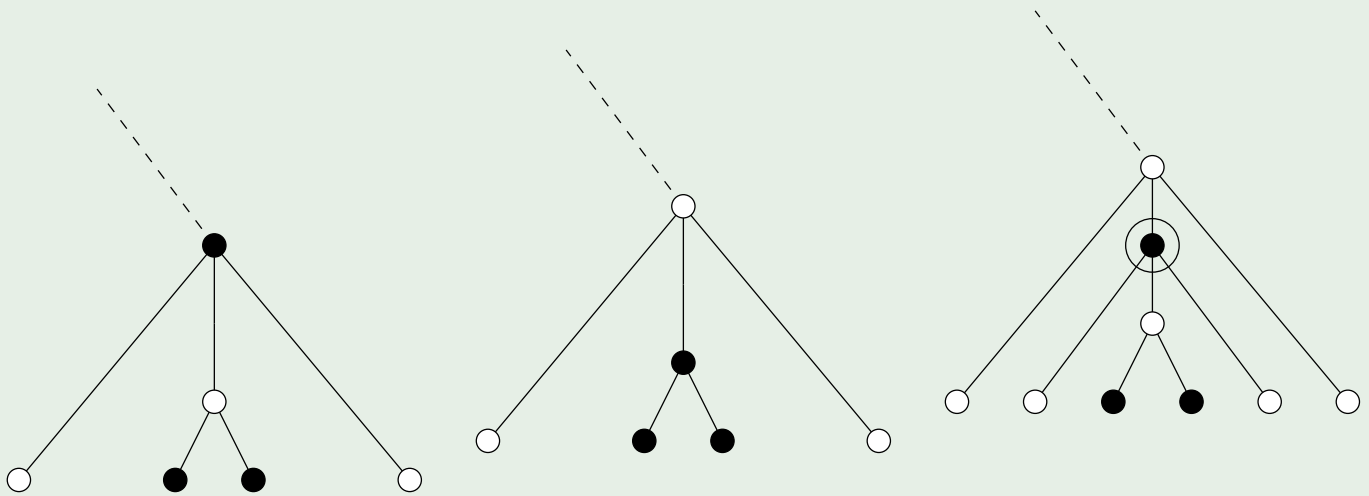


SDEP : 5 10

# Fundamental lemma

11 min

## Example ($\delta = 3$)

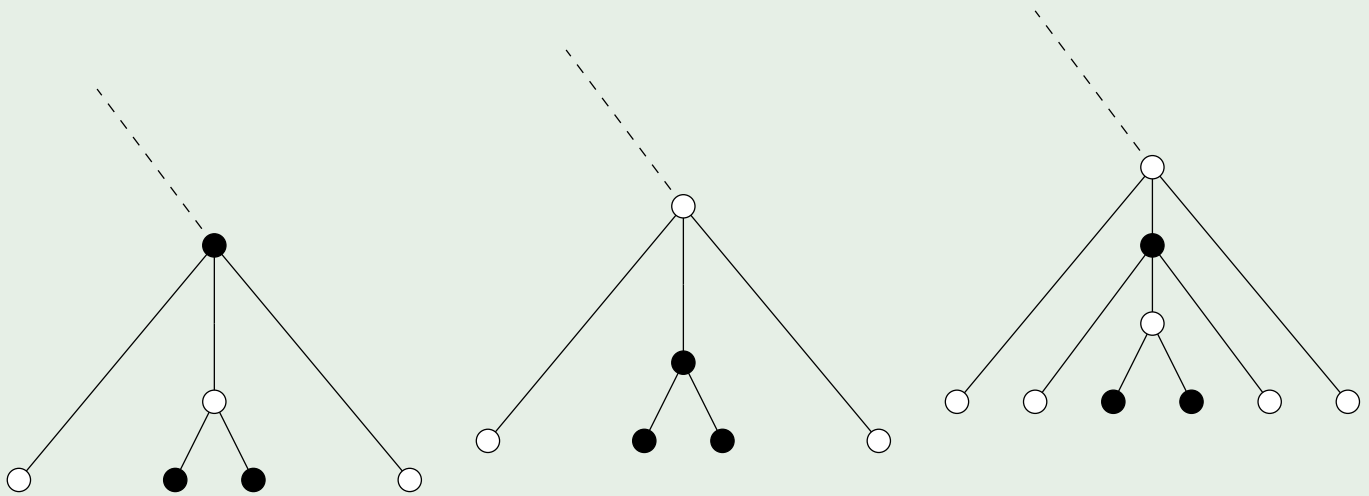


SDEP : 5 10 7

# Fundamental lemma

11 min

## Example ($\delta = 3$)

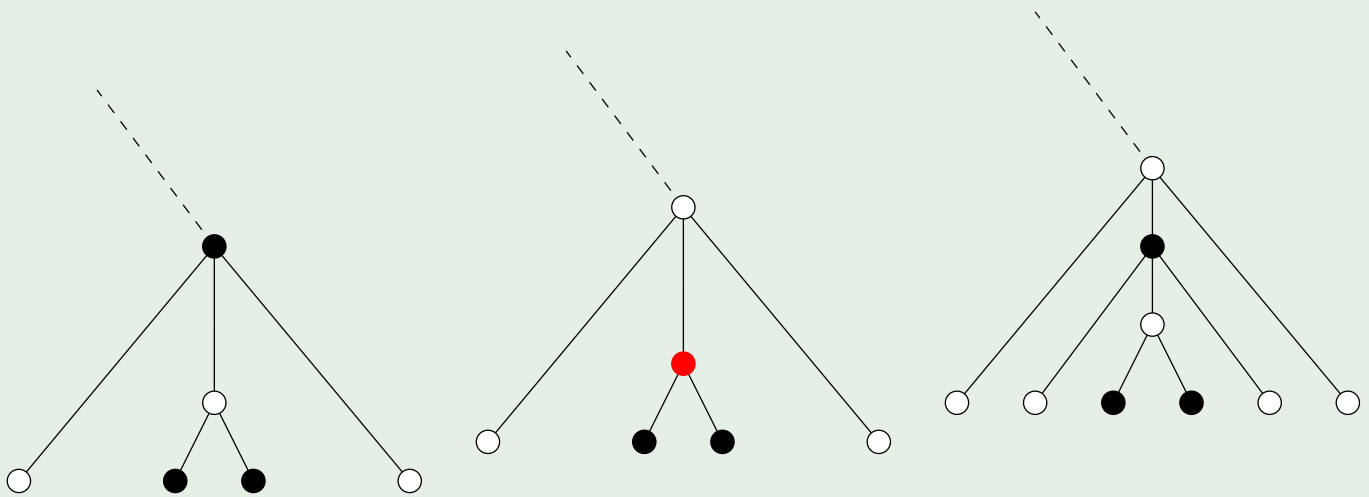


5+0 10+1 7+2

# Fundamental lemma

11 min

## Example ($\delta = 3$)

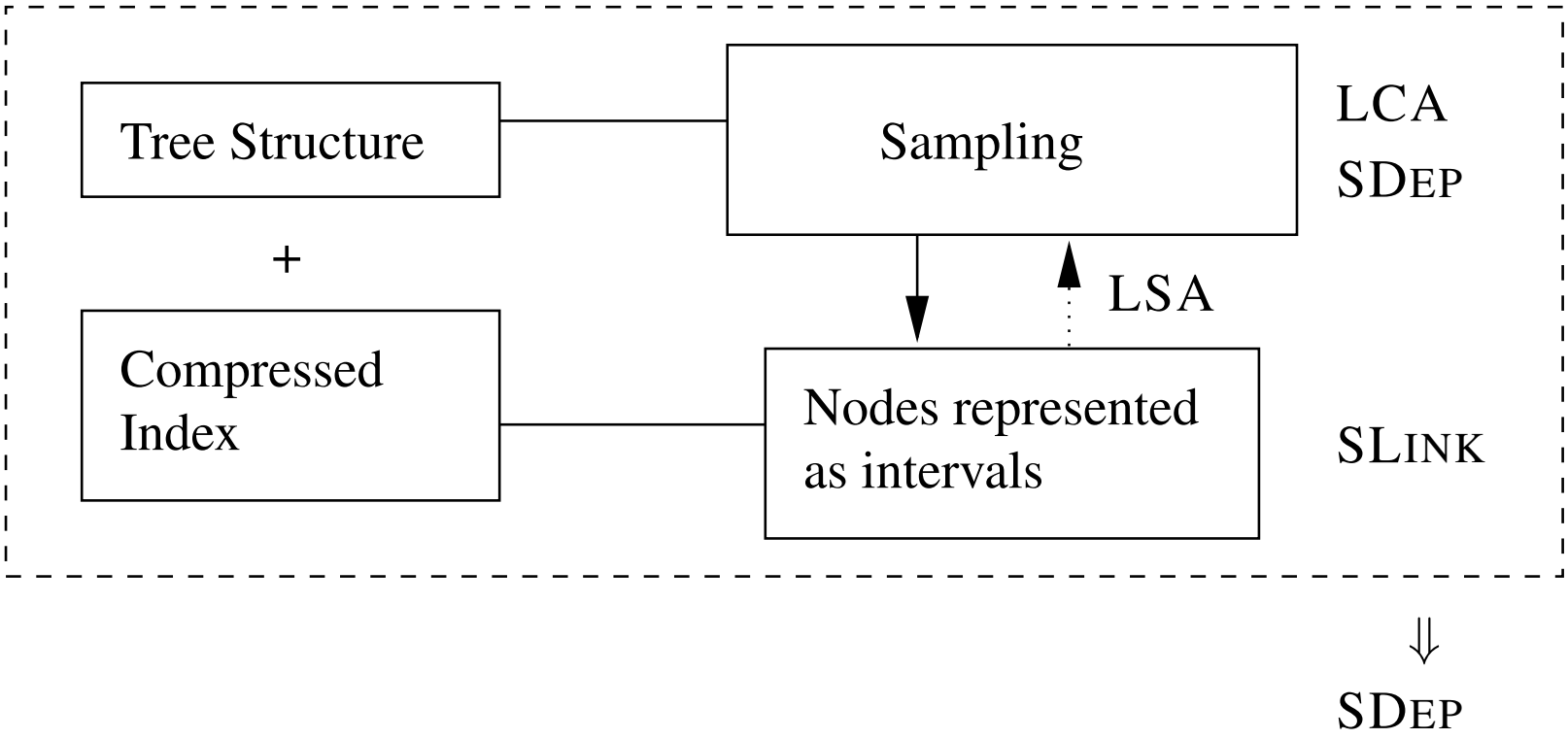


5+0 10+1 7+2

# Entangled Operations



Why is the lemma important ?



# Entangled Operations



The lemma allows us to compute other operations:

- $\textsc{SDep}(v) = \textsc{SDep}(\textsc{LCA}(v, v))$.
- $\textsc{SLink}(v) = \textsc{LCA}(\psi(v_l), \psi(v_r))$,
  $\textsc{SLink}^i(v) = \textsc{LCA}(\psi^i(v_l), \psi^i(v_r))$.
- $\textsc{LCA}(v, v') =$
  $\textsc{LF}(v[0..i-1],$
  $\textsc{LCSA}(\textsc{SLink}^i(v), \textsc{SLink}^i(v')))$,
  for the $i$ in the lemma.

SLink depends on LCA and LCA on SLink.

# Entangled Operations

8 min

The lemma allows us to compute other operations:

- $\textsc{SDep}(v) = \textsc{SDep}(\textsc{LCA}(v, v))$.
- $\textsc{SLink}(v) = \textsc{LCA}(\psi(v_l), \psi(v_r))$,
  $\textsc{SLink}^i(v) = \textsc{LCA}(\psi^i(v_l), \psi^i(v_r))$.
- $\textsc{LCA}(v, v') =$
  $\textsc{LF}(v[0..i-1],$
  $\textsc{LCSA}(\textsc{SLink}^i(v), \textsc{SLink}^i(v')))$,
  for the $i$ in the lemma.

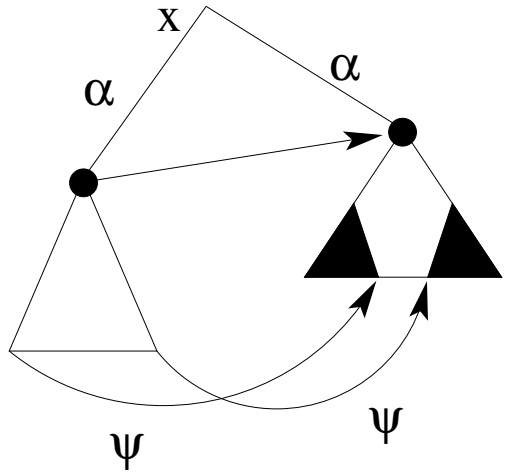
SLink depends on LCA and LCA on SLink.

# Entangled Operations

8 min

The lemma allows us to compute other operations:

- $\textsc{SDep}(v) = \textsc{SDep}(\textsc{LCA}(v, v))$.
- $\textsc{SLink}(v) = \textsc{LCA}(\psi(v_l), \psi(v_r))$,
  $\textsc{SLink}^i(v) = \textsc{LCA}(\psi^i(v_l), \psi^i(v_r))$.
- $\textsc{LCA}(v, v') =$
  $\textsc{LF}(v[0..i-1],$
  $\textsc{LCSA}(\textsc{SLink}^i(v), \textsc{SLink}^i(v')))$,
  for the $i$ in the lemma.



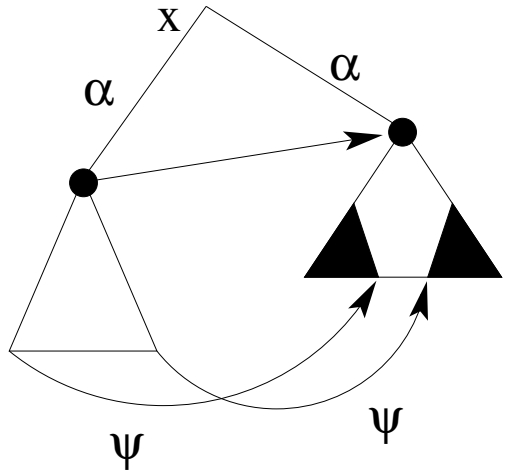

SLink depends on LCA and LCA on SLink.

# Entangled Operations

8 min

The lemma allows us to compute other operations:

- $\text{SDEP}(v) = \text{SDEP}(\text{LCA}(v, v))$.
- $\text{SLINK}(v) = \text{LCA}(\psi(v_l), \psi(v_r))$,
  $\text{SLINK}^i(v) = \text{LCA}(\psi^i(v_l), \psi^i(v_r))$.
- $\text{LCA}(v, v') =$
  $\text{LF}(v[0..i-1],$
  $\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))$,
  for the $i$ in the lemma.

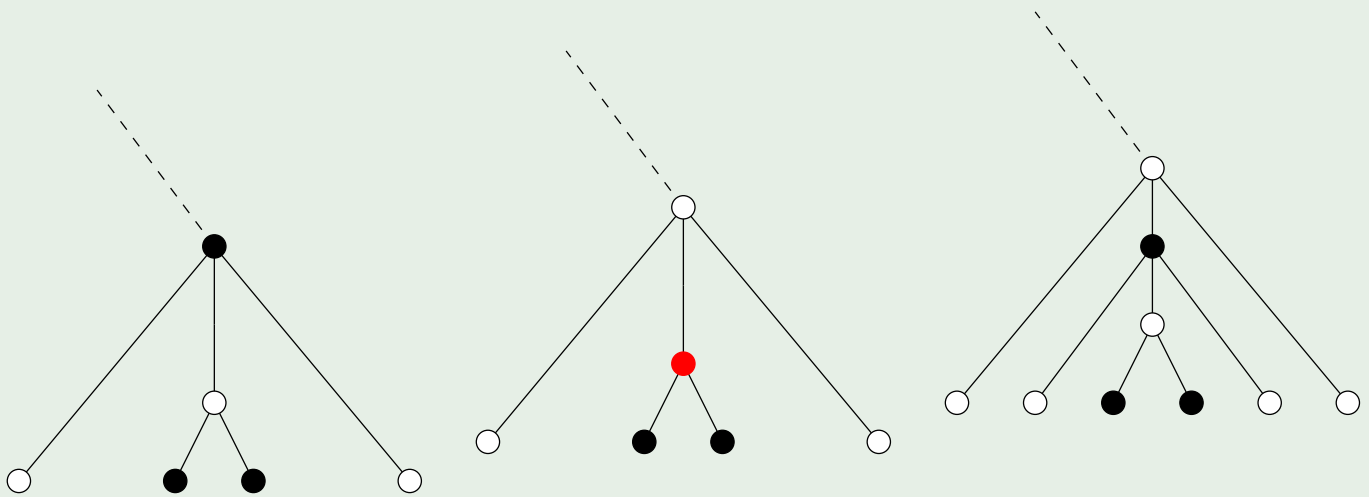



SLINK depends on LCA and LCA on SLINK.

# Entangled Operations

7 min

## Example ($\delta = 3$)

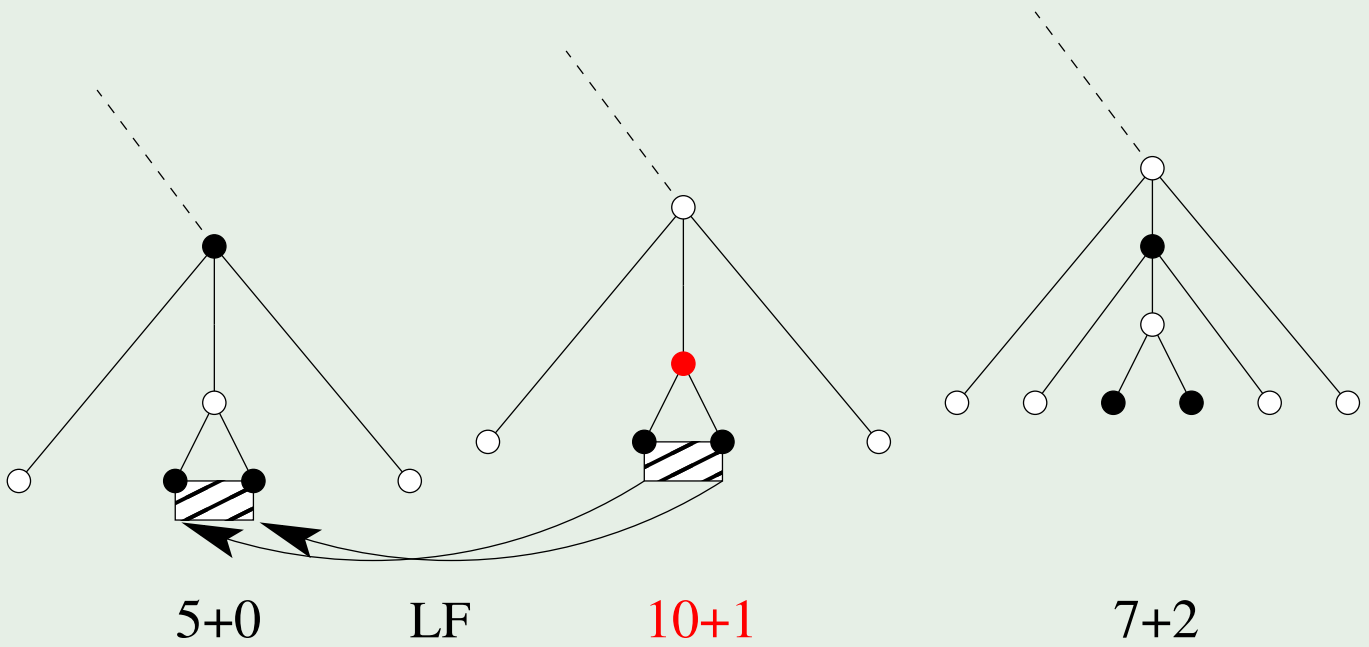


5+0 10+1 7+2

# Entangled Operations

7 min

## Example ($\delta = 3$)

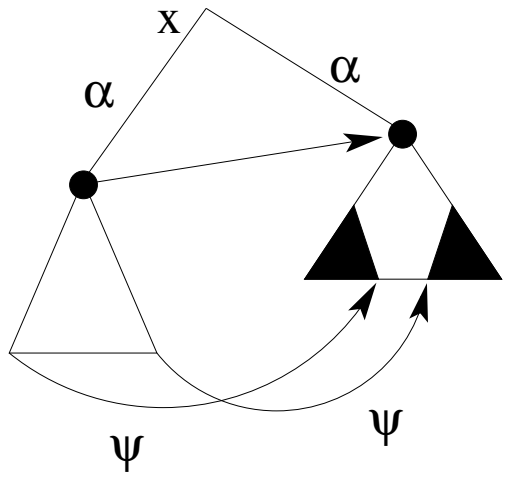


# Entangled Operations

6 min

The lemma allows us to compute other operations:

- $\text{SDEP}(v) = \text{SDEP}(\text{LCA}(v, v))$.
- $\text{SLINK}(v) = \text{LCA}(\psi(v_l), \psi(v_r))$,
  $\text{SLINK}^i(v) = \text{LCA}(\psi^i(v_l), \psi^i(v_r))$.
- $\text{LCA}(v, v') =$
  $\text{LF}(v[0..i-1],$
  $\text{LCSA}(\text{SLINK}^i(v), \text{SLINK}^i(v')))$,
  for the $i$ in the lemma.



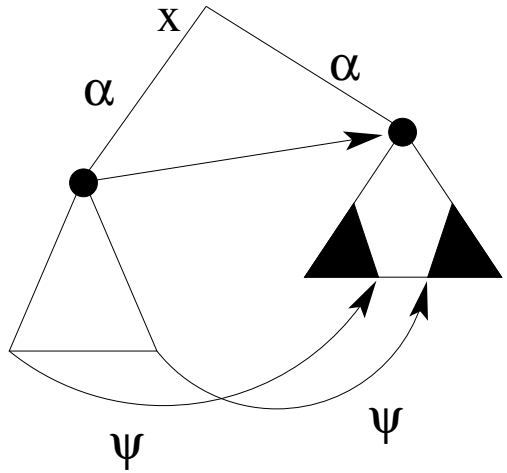

SLINK depends on LCA and LCA on SLINK.

# Entangled Operations

6 min

The lemma allows us to compute other operations:

- $\textsc{SDep}(v) = \textsc{SDep}(\textsc{LCA}(v, v))$.
- $\textsc{SLink}(v) = \textsc{LCA}(\psi(v_l), \psi(v_r))$,
  $\textsc{SLink}^i(v) = \textsc{LCA}(\psi^i(v_l), \psi^i(v_r))$.
- $\textsc{LCA}(v, v') =$
  $\textsc{LF}(v[0..i-1],$
  $\textsc{LCSA}(\textsc{SLink}^i(v), \textsc{SLink}^i(v')))$,
  for the $i$ in the lemma.




SLink depends on LCA and LCA on SLink.

# Breaking the Cycle

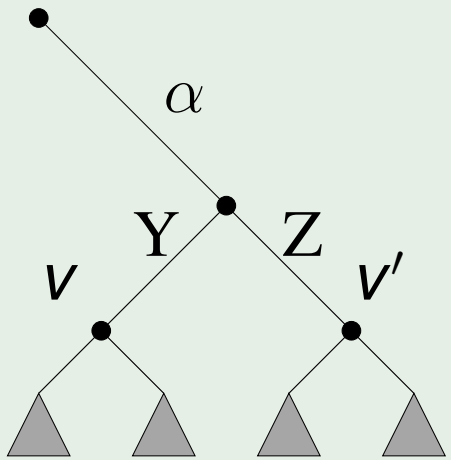
5 min

To avoid this circular dependency we use the next lemma.

## Lemma

$$\mathrm{LCA}(v, v') = \mathrm{LCA}(\min\{v_l, v'_l\}, \max\{v_r, v'_r\})$$

## Example



# Breaking the Cycle

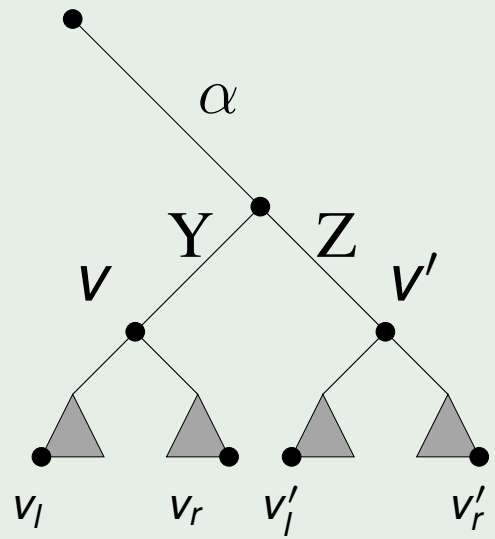
5 min

To avoid this circular dependency we use the next lemma.

## Lemma

$$\mathrm{LCA}(v, v') = \mathrm{LCA}(\min\{v_l, v'_l\}, \max\{v_r, v'_r\})$$

## Example



# Breaking the Cycle

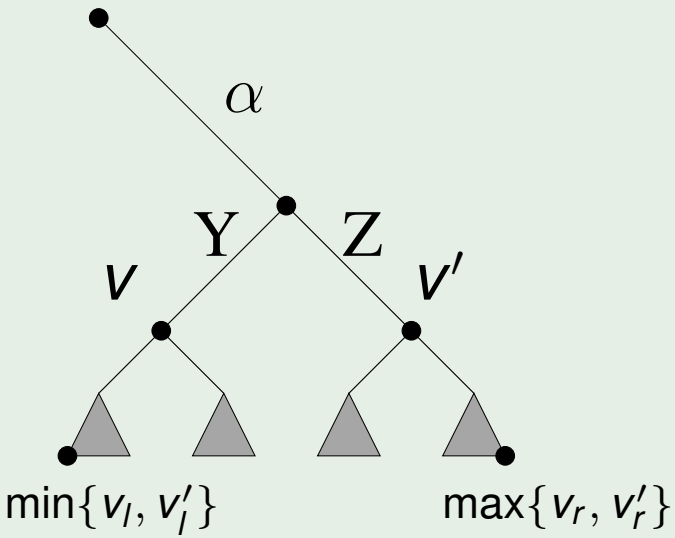
5 min

To avoid this circular dependency we use the next lemma.

## Lemma

$$\mathrm{LCA}(v, v') = \mathrm{LCA}(\min\{v_l, v'_l\}, \max\{v_r, v'_r\})$$

## Example



# Breaking the Cycle

4 min

Hence we can use $\psi$ instead of SLink.
Therefore LCA no longer depends on SLink.
The following operations simplify:

- $\mathrm{SDep}(v) = \mathrm{SDep}(\mathrm{LCA}(v, v)) = \max_{0 \le i < d}\{i + \mathrm{SDep}(\mathrm{LCSA}(\psi^i(v_l), \psi^i(v_r)))\}$.
- $\mathrm{LCA}(v, v') = \mathrm{LF}(v[0..i-1], \mathrm{LCSA}(\psi^i(\min\{v_l, v'_l\}), \psi^i(\max\{v_r, v'_r\})))$, for the $i$ in the lemma.

# Breaking the Cycle

4 min

Hence we can use $\psi$ instead of SLink.
Therefore LCA no longer depends on SLink.
The following operations simplify:

- $\text{SDep}(v) = \text{SDep}(\text{LCA}(v, v)) =$
  $\max_{0 \le i < d}\{i + \text{SDep}(\text{LCSA}(\psi^i(v_l), \psi^i(v_r)))\}$.
- $\text{LCA}(v, v') =$
  $\text{LF}(v[0..i-1],$
  $\text{LCSA}(\psi^i(\min\{v_l, v'_l\}), \psi^i(\max\{v_r, v'_r\})))$,
  for the $i$ in the lemma.

# Breaking the Cycle

4 min

Hence we can use $\psi$ instead of SLink.
Therefore LCA no longer depends on SLink.
The following operations simplify:

- $\text{SDep}(v) = \text{SDep}(\text{LCA}(v, v)) =$
  $\max_{0 \le i < d}\{i + \text{SDep}(\text{LCSA}(\psi^i(v_l), \psi^i(v_r)))\}$.
- $\text{LCA}(v, v') =$
  $\text{LF}(v[0..i-1],$
  $\text{LCSA}(\psi^i(\min\{v_l, v'_l\}), \psi^i(\max\{v_r, v'_r\})))$,
  for the $i$ in the lemma.

# Further Operations

3 min

With these base operations we can also compute:

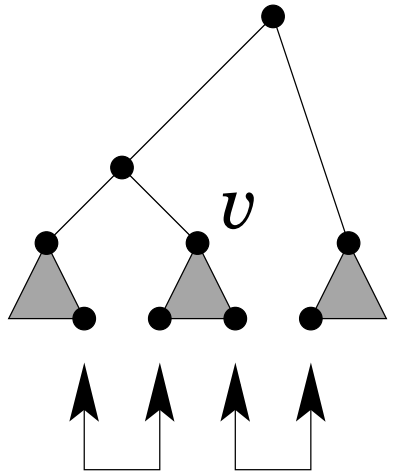
- LETTER$(v, i) =$ SLINK$^i(v)[0] =$ $\psi^i(v_l)[0]$
- PARENT is either LCA$(v_l - 1, v_l)$ or LCA$(v_r, v_r + 1)$, whichever is lowest.

# Further Operations

3 min

With these base operations we can also compute:

- $\textsc{Letter}(v, i) = \textsc{SLink}^i(v)[0] = \psi^i(v_l)[0]$
- $\textsc{Parent}$ is either $\text{LCA}(v_l - 1, v_l)$ or $\text{LCA}(v_r, v_r + 1)$, whichever is lowest.



# Further Operations

1 min

- CHILD can be computed with LETTER and binary searches.
- We can also use the fundamental lemma as $\text{CHILD}(v, X) = \text{LF}(v[0..i-1], \text{CHILD}(\text{SLINK}^i(v), X))$
- The branching is computed over child lists in the sampled tree.
- We proposed a compromise between these approaches.

# Further Operations



- CHILD can be computed with LETTER and binary searches.
- We can also use the fundamental lemma as $\text{CHILD}(v, X) = \text{LF}(v[0..i-1], \text{CHILD}(\text{SLINK}^i(v), X))$
- The branching is computed over child lists in the sampled tree.
- We proposed a compromise between these approaches.

# Further Operations



- CHILD can be computed with LETTER and binary searches.
- We can also use the fundamental lemma as $\text{CHILD}(v, X) = \text{LF}(v[0..i-1], \text{CHILD}(\text{SLINK}^i(v), X))$
- The branching is computed over child lists in the sampled tree.
- We proposed a compromise between these approaches.

# Summary

0 min

We presented a representation of suffix tree that:

- occupies $uH_k + o(u \log \sigma)$ bits.
- supports usual operations in a reasonable time.

# Acknowledgments




- FCT, grant SFRH/BD/12101/2003 of project POCI 2010.
- Fondecyt Grant 1-050493 (Chile).

# Acknowledgments



Thanks for listening.