Lempel Ziv Computation In Small Space (LZ-CISS)

Johannes Fischer  Tomohiro I  Dominik Köppl

Department of Computer Science, TU Dortmund, Germany

30.6.2015
CPM
### Text: aaabaabaaabaa$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Factor</strong></td>
<td>a</td>
<td>aa</td>
<td>b</td>
<td>aabaa</td>
<td>abaa</td>
<td>$</td>
</tr>
<tr>
<td><strong>Coding</strong></td>
<td>a</td>
<td>1,2</td>
<td>b</td>
<td>2,5</td>
<td>3,4</td>
<td>$</td>
</tr>
</tbody>
</table>

### LZ77

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Factor</strong></td>
<td>a</td>
<td>aa</td>
<td>b</td>
<td>aab</td>
<td>aaa</td>
<td>ba</td>
</tr>
<tr>
<td><strong>Coding</strong></td>
<td>a</td>
<td>1,a</td>
<td>b</td>
<td>2,b</td>
<td>2,a</td>
<td>3,a</td>
</tr>
</tbody>
</table>
## Quest for small working space

### LZ77

<table>
<thead>
<tr>
<th>Time</th>
<th>Bits of working space</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(n)$</td>
<td>$3n \lg n$</td>
<td>Goto and Bannai’13</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$2n \lg n$</td>
<td>Kärkkäinen et al.’13</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$n \lg n + O(\sigma \lg n)$</td>
<td>Goto and Bannai’14</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$(1 + \epsilon)n \lg n + O(n)$</td>
<td>this paper</td>
</tr>
</tbody>
</table>

### LZ78

<table>
<thead>
<tr>
<th>Time</th>
<th>Bits of working space</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(n \lg \sigma)$</td>
<td>$O(z \lg z)$</td>
<td>folklore</td>
</tr>
<tr>
<td>$O\left(n + z \frac{\lg \sigma}{\lg \lg \sigma}\right)$</td>
<td>$O(z \lg z)$</td>
<td>Fischer, Gawrychowski’15</td>
</tr>
<tr>
<td>$O\left(\frac{n \lg \sigma}{\lg \sigma \lg \lg \lg n}\right)$</td>
<td>$O\left(n \lg \sigma + n \frac{\lg \lg \sigma n}{\lg \sigma}\right)$</td>
<td>Jansson et al.’15</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$O(n \lg n)$</td>
<td>Nakashima et al.’15</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>$(1 + \epsilon)n \lg n + O(n)$</td>
<td>this paper</td>
</tr>
</tbody>
</table>

$n$: text size  \hspace{1cm}  $\sigma$: alphabet size  \hspace{1cm}  $z$: #factors  \hspace{1cm}  $0 < \epsilon \leq 1$ constant
Suffix Tree

ST of aaabaabaaabaa

Labels
- internal nodes: DFS number
- leaves: text pos. of suffix

Superimposition
- suffix trie
- LZ78 trie subtree of suffix trie
- edge counts how far it got explored.
Suffix Tree

ST of aaabaabaaabaa

Labels
- internal nodes: DFS number
- leaves: text pos. of suffix

Superimposition
- suffix trie superimposed by suffix tree
- LZ78 trie subtree of suffix trie
- edge counts how far it got explored.
Suffix Tree

ST of aaabaabaaaabaa

Labels
- internal nodes: DFS number
- leaves: text pos. of suffix

Superimposition
- suffix trie superimposed by suffix tree
- LZ78 trie subtree of suffix trie
- edge counts how far it got explored.
**Suffix Tree**

ST of aaabaabaaabaa

Labels
- internal nodes: DFS number
- leaves: text pos. of suffix

Superimposition
- suffix trie superimposed by suffix tree
- LZ78 trie subtree of suffix trie
- edge counts how far it got explored.
Add new factor ≡ append new node to trie:

- Classic Trie: Walk down from root.
- Now: Use Level Ancestor Query on ST leaf
- Traverse in $\mathcal{O}(1)$ time.
LZ78 parsing of aaabaabaaabaaabaa:

```
a |aa |b |aab |aaa |ba |a$
```

Witnesses:
3, 5, 18, 10, 7, 18, 4

Definition
Witness: highest ST node below of or equal to LZ78 trie node.
LZ78 parsing of
\texttt{aaabaabaaabaa}:
\texttt{a | aa | b | aab | aaa | ba | a$}

Witnesses:
3, 5, 18, 10, 7, 18, 4

Definition
Witness: highest ST node below of or equal to LZ78 trie node.
LZ78 parsing of
\texttt{aaabaabaaabaa}:
\texttt{a | aa | b | aab | aaa | ba | a$}

**Witnesses:**
3, 5, 18, 10, 7, 18, 4

**Definition**

**Witness:** highest ST node below of or equal to LZ78 trie node.
LZ78 parsing of aaabaabaaabaa:

\[ a \mid aa \mid b \mid aab \mid aaa \mid ba \mid a\$

Witnesses:
3, 5, 18, 10, 7, 18, 4

Definition
Witness: highest ST node below of or equal to LZ78 trie node.
LZ78 parsing of aaabaabaaabaa:

\[
\begin{align*}
\text{a} & | \text{aa} | \text{b} | \text{aab} | \text{aaa} | \text{ba} | \text{a}\$
\end{align*}
\]

Witnesses:

3, 5, 18, 10, 7, 18, 4

Definition

Witness: highest ST node below of or equal to LZ78 trie node.
LZ78 parsing of aaabaabaaabaa:

\[
\begin{align*}
\text{aa} & | \text{aa} & | \text{ba} & | \text{aab} & | \text{aaa} & | \text{ba} & | \text{a} \\
\end{align*}
\]

Witnesses:

3, 5, 18, 10, 7, 18, 4

Definition
Witness: highest ST node below of or equal to LZ78 trie node.
LZ78 parsing of \textit{aaabaabaaabaa}: 
\texttt{a | aa | b | aab | aaa | ba | a$}

Witnesses:
3, 5, 18, 10, 7, 18, 4

Definition
Witness: highest ST node below of or equal to LZ78 trie node.
LZ78 parsing of
\texttt{aaabaabaaabaa}:
\begin{itemize}
  \item a
  \item aa
  \item b
  \item aab
  \item aaa
  \item ba
  \item a$
\end{itemize}

Witnesses:
\begin{itemize}
  \item 3,
  \item 5,
  \item 18,
  \item 10,
  \item 7,
  \item 18,
  \item 4
\end{itemize}

Definition
Witness: highest ST node below of or equal to LZ78 trie node.
LZ78 parsing of $aaabaabaaabaa$:

$\text{a | aa | b | aab | aaa | ba | a\$}$

Witnesses:

$3, 5, 18, 10, 7, 18, 4$

Definition

Witness: highest ST node below of or equal to LZ78 trie node.
LZ78 parsing of \texttt{aaabaabaaabaa}:
\begin{verbatim}
a | aa | b | aab | aaa | ba | a$
\end{verbatim}

Witnesses:
3, 5, 18, 10, 7, 18, 4

Definition
Witness: highest ST node below of or equal to LZ78 trie node.
Witnesses → References

Working Space
Witnesses → References

\[ \begin{array}{cccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\end{array} \]

\[ \begin{array}{cccccccc}
3 & 5 & 18 & 10 & 7 & 18 & 4 \\
\end{array} \]

Working Space

\[ \begin{array}{cccccccc}
3 & 4 & 5 & 7 & 10 & 18 \\
\end{array} \]
Witnesses → References

Working Space
Witnesses → References

Select factor

1 2 3 4 5 6 7

Lookup previous Factor

Working Space
Witnesses → References

Working Space

parent(5) = 3

parent(3) = 1

parent(4) = 5

parent(10) = 7

parent(18) = 4

parent(1) = 2
Witnesses → References

Referenced Factors

Working Space
Two passes over the suffix tree:

1 Pass:
- Traverse from every leaf to the root
- Mark visited nodes
- Already marked nodes \(\equiv\) some reference
- These nodes \textit{witness} references

2 Pass:
- Same procedure
- We know \textit{witnesses} already!
- Which leaf discovers a \textit{witness} first?
LZ77 parsing of
\texttt{aaabaabaaaabaa}:
\texttt{a|aa|b|aabaa|abaa|}.

Starting positions of factors (≡ leaf labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

Definition
Witness: already visited
definition
node accessed by LZ leaf.

\texttt{LZ77 parsing of}
\texttt{aaabaabaaaabaa}:
\texttt{a|aa|b|aabaa|abaa|}.

\textbf{Witnesses:}
5, 10, 14

\textbf{Definition}
Witness: already visited
definition
node accessed by LZ leaf.
LZ77 parsing of
aaabaabaaabaa:
\[ a | aa | b | aabaa | abaa | $. \]

Starting positions of factors (≡ leaf labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

Definition
Witness: already visited
node accessed by LZ leaf.
LZ77 parsing of
aaabaabaaabaa : 
a | aa | b | aabaa | abaa | $.

Starting positions of factors (≡ leaf labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

Definition
Witness: already visited
node accessed by LZ leaf.
LZ77 parsing of
\texttt{aaabaabaaabaa} :
\texttt{a |aa |b |aabaa |abaa |$}.

Starting positions of factors (≡ leaf labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

\textbf{Definition}
\textbf{Witness: already visited node accessed by LZ leaf.}
LZ77 parsing of
\text{aaabaabaaabaa} : \ a|aa|b|aabaa|abaa|.$.

Starting positions of factors (≡ leaf labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

Definition
Witness: already visited node accessed by LZ leaf.
LZ77 parsing of

```
aaabaabaaabaa
```

: 
```
a | aa | b | aabaa | abaa |
```

Starting positions of factors (≡ leaf labels):

1, 2, 4, 5, 10, 14

Witnesses:

5, 10, 14

Definition

Witness: already visited node accessed by LZ leaf.
LZ77 parsing of
aaabaabaaabaaabaa:

\[
\begin{align*}
\texttt{a} & | \texttt{aa} | \texttt{b} | \texttt{aabaa} | \texttt{abaa} | \texttt{$}.
\end{align*}
\]

Starting positions of factors \((\equiv\) leaf labels\):  
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

**Definition**

**Witness:** already visited node accessed by LZ leaf.
LZ77 parsing of
aaabaabaaabaa:
a |aa |b |aabaa |abaa |$.

Starting positions of factors (≡ leaf labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

Definition
Witness: already visited
node accessed by LZ leaf.
LZ77 parsing of $aaabaabaabaaabaa$:

\[ a \mid aa \mid b \mid aabaa \mid abaa \mid . \]

Starting positions of factors (≡ leaf labels):

\[ 1, 2, 4, 5, 10, 14 \]

Witnesses:

\[ 5, 10, 14 \]

Definition

Witness: already visited node accessed by LZ leaf.
LZ77 parsing of $aaabaabaaabaa$:

$$a | aa | b | aabaa | abaa | .$$

Starting positions of factors (≡ leaf labels):

$$1, 2, 4, 5, 10, 14$$

Witnesses:

$$5, 10, 14$$

Definition

Witness: already visited node accessed by LZ leaf.
LZ77 parsing of
\texttt{aaabaabaaaabaa} : 
\texttt{a |aa |b |aabaa |abaa |\$}.

Starting positions of 
factors (≡ leaf labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

Definition
Witness: already visited
node accessed by LZ leaf.
LZ77 parsing of $aaabaabaaabaa$:
$a | aa | b | aabaa | abaa | $.

Starting positions of factors ($\equiv$ leaf labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

Definition
Witness: already visited node accessed by LZ leaf.
LZ77 parsing of 
\texttt{aaabaabaaabaa}: 
\texttt{a |aa |b |aabaa |abaa |\$}.

Starting positions of factors (\equiv leaf labels): 
1, 2, 4, 5, 10, 14

Witnesses: 
5, 10, 14

\textbf{Definition} 
\textit{Witness: already visited node accessed by LZ leaf.}
LZ77 parsing of
aaabaabaaabaa : a |aa| b |aabaa |abaa |$. 

Starting positions of factors (≡ leaf labels): 1, 2, 4, 5, 10, 14

Witnesses: 5, 10, 14

Definition
Witness: already visited node accessed by LZ leaf.
LZ77 parsing of
\texttt{aaabaabaaabaa} :  \\
\texttt{a|aa|b|aabaa|abaa|\$.}

Starting positions of factors (≡ leaf labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

Definition
Witness: already visited
node accessed by LZ leaf.
LZ77 parsing of 
\texttt{aaabaabaaabaa} : 
a |aa |b |aabaa |abaa |.$.

Starting positions of factors (≡ leaf labels): 
1, 2, 4, 5, 10, 14

Witnesses: 
5, 10, 14

Definition
Witness: already visited node accessed by LZ leaf.
Two passes over the suffix tree:

1 Pass:
- Traverse from every leaf to the root
- Mark visited nodes
- Already marked nodes $\equiv$ some reference
- These nodes witness references

2 Pass:
- Same procedure
- We know witnesses already!
- Which leaf discovers a witness first?
LZ77 parsing of $T = \text{aaabaabaaabaa as } a \mid a a \mid b \mid aabaa \mid abaa \mid \$$.  

Starting Positions of Factors (≡ Leaf Labels): 1, 2, 4, 5, 10, 14  

Witnesses: 5, 10, 14  

Map text positions to nodes:
LZ77 parsing of $T = \text{aaabaabaaabaa}$ as $a \vert \text{aa} \vert b \vert \text{aabaa} \vert \text{abaa} \vert \$.$

Starting Positions of Factors (≡ Leaf Labels): 1, 2, 4, 5, 10, 14

Witnesses: 5, 10, 14

Map text positions to nodes:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>5</td>
<td>10</td>
<td>14</td>
<td>10</td>
<td>14</td>
<td>5</td>
<td>10</td>
<td>5</td>
<td>14</td>
<td>10</td>
<td>14</td>
<td></td>
</tr>
</tbody>
</table>
LZ77 parsing of $T = \texttt{aaabaabaaabaa}$ as $a | aa | b | aabaa | abaa |$.

Starting Positions of Factors (≡ Leaf Labels): 1, 2, 4, 5, 10, 14

Witnesses: 5, 10, 14

Map text positions to nodes:

\[
\begin{array}{cccccccc}
1 & 2 & 3 & 5 & 10 & 5 & 14 & 10 & 14 \\
5 & 10 & 5 & 14 & 10 & 14 \\
\end{array}
\]
LZ77 parsing of $T = \text{aaabaabaaabaa}$ as
a |aa |b |aabaa |abaa |$.

Starting Positions of Factors ($\equiv$ Leaf Labels):
$1, 2, 4, 5, 10, 14$

Witnesses:
$5, 10, 14$

Map text positions to nodes:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>10</th>
<th>14</th>
</tr>
</thead>
</table>
LZ77 parsing of $T = \text{aaabaabaaabaa}$ as $a | aa | b | aabaa | abaa | \$. 

Starting Positions of Factors (≡ Leaf Labels): 1, 2, 4, 5, 10, 14

Witnesses: 5, 10, 14

Map text positions to nodes:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>10</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>5</td>
<td>14</td>
<td>10</td>
<td>14</td>
<td></td>
</tr>
</tbody>
</table>
LZ77 parsing of $T = \text{aaabaabaaabaa}$ as
\[a | aa | b | aabaa | abaa | $.\]

Starting Positions of Factors (≡ Leaf Labels):
1, 2, 4, 5, 10, 14

Witnesses:
5, 10, 14

Map text positions to nodes:
<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>10</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>5</td>
<td>14</td>
<td>10</td>
<td>14</td>
</tr>
</tbody>
</table>
LZ77 parsing of $T = a a b a a b a a a b a a a$ as $a | a a | b | a a b a a | a b a a | .$

Starting Positions of Factors (≡ Leaf Labels): $1, 2, 4, 5, 10, 14$

Witnesses: $5, 10, 14$

Map text positions to nodes:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>5</td>
<td>14</td>
<td>10</td>
<td>14</td>
</tr>
</tbody>
</table>
LZ77 parsing of $T = \text{aaabaabaaabaa as a | aa | b | aabaa | abaa | }$. 

Starting Positions of Factors (≡ Leaf Labels): 1, 2, 4, 5, 10, 14

Witnesses: 5, 10, 14

Map text positions to nodes:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>10</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>5</td>
<td>14</td>
<td>10</td>
<td>14</td>
</tr>
</tbody>
</table>
LZ77 parsing of $T = aaabaabaaabaaabaa$ as $a | aa | b | aabaa | abaa | \$.$

Starting Positions of Factors (≡ Leaf Labels): 1, 2, 4, 5, 10, 14

Witnesses: 5, 10, 14

Map text positions to nodes:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>10</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>5</td>
<td>14</td>
<td>10</td>
<td>14</td>
</tr>
</tbody>
</table>
LZ77 parsing of $T = \text{aaabaabaaabaa}$ as $a \mid \text{aa} \mid b \mid \text{aabaa} \mid \text{abaa} \mid \$$. 

Starting Positions of Factors (≡ Leaf Labels): 1, 2, 4, 5, 10, 14

Witnesses: 5, 10, 14

Map text positions to nodes:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>10</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>5</td>
<td>14</td>
<td>10</td>
<td>14</td>
<td></td>
</tr>
</tbody>
</table>
Find Reference Starting Positions

Match factors with referencing text positions.

- Take witness of factor.
- Search first occurrence of witness.
- We found the referring text position!

By this we get:

- Factor at text-pos 2 refers to pos 1.
- Factor at text-pos 5 refers to pos 2.
Find Reference Starting Positions

Text positions

1 2 2 3 5 10
5 10 5 14 10 14

Match factors with referencing text positions.
- Take witness of factor.
- Search first occurrence of witness.
- We found the referring text position!

By this we get:
- Factor at text-pos 2 refers to pos 1.
- Factor at text-pos 5 refers to pos 2.
Find Reference Starting Positions

Match factors with referencing text positions.

- Take **witness** of factor.
- Search **first occurrence of witness**.
- We found the referring text position!

By this we get:

- Factor at text-pos 2 refers to pos 1.
- Factor at text-pos 5 refers to pos 2.
Tricks & Techniques
Main Idea

Use Suffix Tree (ST):
- Take leaves by text position.
- String depth reveals factor length.
- Compute references indirectly by nodes called witnesses.
# Auxiliary Data Structures

Build a lightweight Suffix Tree (ST) with:
- Enhanced Suffix Array
- DFUDS tree topology
- MinMax tree for ST navigation

<table>
<thead>
<tr>
<th>DS</th>
<th>space in bits</th>
<th>constr. time</th>
<th>constr. space</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA</td>
<td>$n \lg n$</td>
<td>$\mathcal{O}(n/\epsilon^2)$</td>
<td>$(1 + \epsilon)n \lg n$</td>
<td>Kärkkäinen et al. ’06</td>
</tr>
<tr>
<td>LCP</td>
<td>$2n + o(n)$</td>
<td>$\mathcal{O}(n)$</td>
<td>$\mathcal{O}(1)$</td>
<td>Välimäki et al. ’09</td>
</tr>
<tr>
<td>DFUDS</td>
<td>$4n + 4$</td>
<td>$\mathcal{O}(n)$</td>
<td>$n + o(n)$</td>
<td>Ohlebusch et al. ’10</td>
</tr>
<tr>
<td>MinMax</td>
<td>$o(n)$</td>
<td>$o(n)$</td>
<td>$o(n)$</td>
<td>Navarro,Sadakane’14</td>
</tr>
<tr>
<td>RMQ</td>
<td>$2n + o(n)$</td>
<td>$\mathcal{O}(n)$</td>
<td>$n + o(n)$</td>
<td>Fischer’10</td>
</tr>
<tr>
<td>total</td>
<td>$n \lg n + \mathcal{O}(n)$</td>
<td>$\mathcal{O}(n)$</td>
<td>$(1 + \epsilon)n \lg n + \mathcal{O}(n)$</td>
<td>this paper</td>
</tr>
</tbody>
</table>

result  
$\mathcal{O}(n)$  
$(1 + \epsilon)n \lg n + \mathcal{O}(n)$  
this paper
Two heavyweight arrays

SA  ST string depth.
   + LCP: Factor length
ISA  fetch ST leaf
   - LZ scans text linear from left to right
   - ST leaves in ISA order = LZ parsing order

Problem

Cannot store SA, ISA and output in \((1 + \epsilon)n \lg n\) bits!
But with a DS of Munro et al.'12:

**Theorem**

*Given a permutation $A[1..n]$, $A$ uses $n \log n$ bits.

The array+inverse of $A$

- answers $A^{-1}$ in $O(1/\epsilon)$ time.
- uses additional $\epsilon n \log n$ bits
- is built in $O(n)$ time.*
Managing Space

Store two arrays in \((1 + \epsilon)n \lg n\) bits:

- \(n \lg n\) bits for
  - SA (in the beginning)
  - ISA (invert SA)
  - storing witnesses (indirect references)

- \(\epsilon n \lg n\) bits for
  - array+inverse: SA with \(\mathcal{O}(1/\epsilon)\) access time
  - helper array for comparisons
Summary

Theorem

Given text $T$ of length $n$. LZ77 and LZ78 of $T$ can be computed
- with $O(n)$ time
- with $(1 + \epsilon)n \lg n + O(n)$ bits space
- without extra output space! (see paper)

Techniques used:
- Succinct, but fast ST.
- Storing SA and ISA in $(1 + \epsilon)n \lg n$ bits.
- Indirect Matching (witnesses).

Thank you for listening. Any questions are welcome!
Summary

Theorem
Given text $T$ of length $n$. LZ77 and LZ78 of $T$ can be computed
- with $O(n)$ time
- with $(1 + \epsilon)n \lg n + O(n)$ bits space
- without extra output space! (see paper)

Techniques used:
- Succinct, but fast ST.
- Storing SA and ISA in $(1 + \epsilon)n \lg n$ bits.
- Indirect Matching (witnesses).

Thank you for listening. Any questions are welcome!