Balancing Straight-Line Programs

Moses Ganardi, Artur Jeż and Markus Lohrey

Universität Siegen

October 10, 2019

Straight-Line Programs

A straight-line program (SLP) is basically a context-free grammar in Chomsky normal form that produces a single string.

Straight-Line Programs

A straight-line program (SLP) is basically a context-free grammar in Chomsky normal form that produces a single string.

Definition (Straight-line program – SLP)

An SLP over the alphabet Γ is a sequence of productions (grammar rules)

$$\mathcal{G} = \langle X_i \to \alpha_i \mid 1 \leq i \leq n \rangle,$$

where either $\alpha_i \in \Gamma$ or $\alpha_i = X_j X_k$ for some $j, k > i$.

Straight-Line Programs

A straight-line program (SLP) is basically a context-free grammar in Chomsky normal form that produces a single string.

Definition (Straight-line program – SLP)

An SLP over the alphabet Γ is a sequence of productions (grammar rules)

$$\mathcal{G} = \langle X_i \to \alpha_i \mid 1 \leq i \leq n \rangle,$$

where either $\alpha_i \in \Gamma$ or $\alpha_i = X_j X_k$ for some $j, k > i$.

The unique string derived from X_i is denoted by $[\![X_i]\!]_{\mathcal{G}}$.

Straight-Line Programs

A straight-line program (SLP) is basically a context-free grammar in Chomsky normal form that produces a single string.

Definition (Straight-line program – SLP)

An SLP over the alphabet Γ is a sequence of productions (grammar rules)

$$\mathcal{G} = \langle X_i \to \alpha_i \mid 1 \le i \le n \rangle,$$

where either $\alpha_i \in \Gamma$ or $\alpha_i = X_j X_k$ for some $j, k > i$.

The unique string derived from X_i is denoted by $[\![X_i]\!]_{\mathcal{G}}$.

The string defined by $\mathcal{G}$ is $[\![\mathcal{G}]\!] = [\![X_1]\!]_{\mathcal{G}}$.

Straight-Line Programs

A straight-line program (SLP) is basically a context-free grammar in Chomsky normal form that produces a single string.

Definition (Straight-line program – SLP)

An SLP over the alphabet Γ is a sequence of productions (grammar rules)

$$\mathcal{G} = \langle X_i \to \alpha_i \mid 1 \leq i \leq n \rangle,$$

where either $\alpha_i \in \Gamma$ or $\alpha_i = X_j X_k$ for some $j, k > i$.

The unique string derived from X_i is denoted by $[\![X_i]\!]_{\mathcal{G}}$.

The string defined by $\mathcal{G}$ is $[\![\mathcal{G}]\!] = [\![X_1]\!]_{\mathcal{G}}$.

The size of $\mathcal{G}$ is $|\mathcal{G}| = n$.

Straight-Line Programs

A straight-line program (SLP) is basically a context-free grammar in Chomsky normal form that produces a single string.

Definition (Straight-line program – SLP)

An SLP over the alphabet Γ is a sequence of productions (grammar rules)

$$\mathcal{G} = \langle X_i \to \alpha_i \mid 1 \leq i \leq n \rangle,$$

where either $\alpha_i \in \Gamma$ or $\alpha_i = X_j X_k$ for some $j, k > i$.

The unique string derived from X_i is denoted by $[\![X_i]\!]_{\mathcal{G}}$.

The string defined by $\mathcal{G}$ is $[\![\mathcal{G}]\!] = [\![X_1]\!]_{\mathcal{G}}$.

The size of $\mathcal{G}$ is $|\mathcal{G}| = n$.

The depth of $\mathcal{G}$ (depth($\mathcal{G}$)) is the height of the derivation tree of $\mathcal{G}$.

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2}$ for $1 \leq i \leq 4, \quad X_5 \to b, \quad X_6 \to a\rangle.$

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2}$ for $1 \le i \le 4, \quad X_5 \to b, \quad X_6 \to a\rangle$.

$$X_1 \;=$$

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2} \text{ for } 1 \leq i \leq 4, \quad X_5 \to b, \quad X_6 \to a \rangle.$

$$X_1 = X_2X_3$$

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2} \text{ for } 1 \le i \le 4, \quad X_5 \to b, \quad X_6 \to a\rangle.$

$$\begin{aligned} X_1 &= X_2X_3 \\ &= X_3X_4X_4X_5 \end{aligned}$$

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2} \text{ for } 1 \leq i \leq 4, \quad X_5 \to b, \quad X_6 \to a \rangle.$

$$\begin{aligned} X_1 &= X_2X_3 \\ &= X_3X_4X_4X_5 \\ &= X_4X_5X_5X_6X_5X_6b \end{aligned}$$

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2}$ for $1 \le i \le 4, \quad X_5 \to b, \quad X_6 \to a\rangle.$

$$\begin{aligned} X_1 &= X_2X_3 \\ &= X_3X_4X_4X_5 \\ &= X_4X_5X_5X_6X_5X_6b \\ &= X_5X_6bbabab \end{aligned}$$

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2} \text{ for } 1 \le i \le 4, \quad X_5 \to b, \quad X_6 \to a\rangle.$

$$\begin{aligned} X_1 &= X_2X_3 \\ &= X_3X_4X_4X_5 \\ &= X_4X_5X_5X_6X_5X_6b \\ &= X_5X_6bbabab \\ &= babbabab \end{aligned}$$

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2}$ for $1 \le i \le 4, \quad X_5 \to b, \quad X_6 \to a\rangle$.

$$\begin{aligned} X_1 &= X_2X_3 \\ &= X_3X_4X_4X_5 \\ &= X_4X_5X_5X_6X_5X_6b \\ &= X_5X_6bbabab \\ &= babbabab \ = \ [\![\mathcal{G}]\!] \end{aligned}$$

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2} \text{ for } 1 \le i \le 4, \quad X_5 \to b, \quad X_6 \to a\rangle.$

$$\begin{aligned} X_1 &= X_2X_3 \\ &= X_3X_4X_4X_5 \\ &= X_4X_5X_5X_6X_5X_6b \\ &= X_5X_6bbabab \\ &= babbabab \;=\; [\![\mathcal{G}]\!] \end{aligned}$$

$$\boxed{\qquad\qquad X_1 \qquad\qquad}$$

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2} \text{ for } 1 \leq i \leq 4, \quad X_5 \to b, \quad X_6 \to a \rangle.$

$$
\begin{aligned}
X_1 &= X_2X_3 \\
&= X_3X_4X_4X_5 \\
&= X_4X_5X_5X_6X_5X_6b \\
&= X_5X_6bbabab \\
&= babbabab \;=\; [\![\mathcal{G}]\!]
\end{aligned}
$$

X_1	
X_2	X_3

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2} \text{ for } 1 \leq i \leq 4, \quad X_5 \to b, \quad X_6 \to a \rangle.$

$$
\begin{aligned}
X_1 &= X_2X_3 \\
&= X_3X_4X_4X_5 \\
&= X_4X_5X_5X_6X_5X_6b \\
&= X_5X_6bbabab \\
&= babbabab \; = \; [\![\mathcal{G}]\!]
\end{aligned}
$$

<table>
<tr><td colspan="4">X_1</td></tr>
<tr><td colspan="2">X_2</td><td colspan="2">X_3</td></tr>
<tr><td>X_3</td><td>X_4</td><td>X_4</td><td>X_5</td></tr>
</table>

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2}$ for $1 \leq i \leq 4, \quad X_5 \to b, \quad X_6 \to a\rangle$.

$$
\begin{aligned}
X_1 &= X_2X_3 \\
&= X_3X_4X_4X_5 \\
&= X_4X_5X_5X_6X_5X_6b \\
&= X_5X_6bbabab \\
&= babbabab \;=\; [\![\mathcal{G}]\!]
\end{aligned}
$$

<table>
<tr><td colspan="8">X_1</td></tr>
<tr><td colspan="5">X_2</td><td colspan="3">X_3</td></tr>
<tr><td colspan="3">X_3</td><td colspan="2">X_4</td><td colspan="2">X_4</td><td>X_5</td></tr>
<tr><td colspan="2">X_4</td><td>X_5</td><td>X_5</td><td>X_6</td><td>X_5</td><td>X_6</td><td>b</td></tr>
</table>

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2} \text{ for } 1 \leq i \leq 4, \quad X_5 \to b, \quad X_6 \to a \rangle.$

$$
\begin{aligned}
X_1 &= X_2X_3 \\
&= X_3X_4X_4X_5 \\
&= X_4X_5X_5X_6X_5X_6b \\
&= X_5X_6bbabab \\
&= babbabab \;=\; [\![\mathcal{G}]\!]
\end{aligned}
$$

<table>
<tr><td colspan="8">X_1</td></tr>
<tr><td colspan="5">X_2</td><td colspan="3">X_3</td></tr>
<tr><td colspan="3">X_3</td><td colspan="2">X_4</td><td colspan="2">X_4</td><td>X_5</td></tr>
<tr><td colspan="2">X_4</td><td>X_5</td><td>X_5</td><td>X_6</td><td>X_5</td><td>X_6</td><td>b</td></tr>
<tr><td>X_5</td><td>X_6</td><td>b</td><td>b</td><td>a</td><td>b</td><td>a</td><td>b</td></tr>
</table>

Example of an SLP

$\mathcal{G} = \langle X_i \to X_{i+1}X_{i+2}$ for $1 \le i \le 4, \quad X_5 \to b, \quad X_6 \to a\rangle$.

$$\begin{aligned} X_1 &= X_2X_3 \\ &= X_3X_4X_4X_5 \\ &= X_4X_5X_5X_6X_5X_6b \\ &= X_5X_6bbabab \\ &= babbabab \;=\; [\![\mathcal{G}]\!] \end{aligned}$$

<table>
<tr><td colspan="8">X_1</td></tr>
<tr><td colspan="5">X_2</td><td colspan="3">X_3</td></tr>
<tr><td colspan="3">X_3</td><td colspan="2">X_4</td><td colspan="2">X_4</td><td>X_5</td></tr>
<tr><td colspan="2">X_4</td><td>X_5</td><td>X_5</td><td>X_6</td><td>X_5</td><td>X_6</td><td>b</td></tr>
<tr><td>X_5</td><td>X_6</td><td>b</td><td>b</td><td>a</td><td>b</td><td>a</td><td>b</td></tr>
<tr><td>b</td><td>a</td><td>b</td><td>b</td><td>a</td><td>b</td><td>a</td><td>b</td></tr>
</table>

The main result

Theorem (Charikar et al. 2002, Rytter 2004)

From a given SLP $\mathcal{G}$ of size n such that $[\![\mathcal{G}]\!]$ has length N, one can construct in time $\mathcal{O}(n \cdot \log N)$ an SLP $\mathcal{G}'$ such that:

- $[\![\mathcal{G}']\!] = [\![\mathcal{G}]\!]$
- $|\mathcal{G}'| \in \mathcal{O}(n \cdot \log N)$
- $\mathrm{depth}(\mathcal{G}') \in \mathcal{O}(\log N)$

The main result

Theorem (Charikar et al. 2002, Rytter 2004)

From a given SLP $\mathcal{G}$ of size n such that $[\![\mathcal{G}]\!]$ has length N, one can construct in time $\mathcal{O}(n \cdot \log N)$ an SLP $\mathcal{G}'$ such that:

- $[\![\mathcal{G}']\!] = [\![\mathcal{G}]\!]$
- $|\mathcal{G}'| \in \mathcal{O}(n \cdot \log N)$
- $\mathrm{depth}(\mathcal{G}') \in \mathcal{O}(\log N)$

Our main result

From a given SLP $\mathcal{G}$ of size n such that $[\![\mathcal{G}]\!]$ has length N, one can construct in time $\mathcal{O}(n)$ an SLP $\mathcal{G}'$ such that:

- $[\![\mathcal{G}']\!] = [\![\mathcal{G}]\!]$
- $|\mathcal{G}'| \in \mathcal{O}(n)$
- $\mathrm{depth}(\mathcal{G}') \in \mathcal{O}(\log N)$

The random access problem

Random access queries

For a string $w \in \Sigma^*$ of length N, a random access query gets a position $i \in [1, N]$ and returns the i-th symbol of w.

The random access problem

Random access queries

For a string $w \in \Sigma^*$ of length N, a random access query gets a position $i \in [1, N]$ and returns the i-th symbol of w.

Theorem (Bille et al. 2015)

From a given SLP $\mathcal{G}$ of size n such that $w := [\![\mathcal{G}]\!]$ has length N, one can compute in time $\mathcal{O}(n)$ a data structure of size $\mathcal{O}(n)$ that allows to answer random access queries for w in time $\mathcal{O}(\log N)$.

The random access problem

Random access queries

For a string $w \in \Sigma^*$ of length N, a random access query gets a position $i \in [1, N]$ and returns the i-th symbol of w.

Theorem (Bille et al. 2015)

From a given SLP $\mathcal{G}$ of size n such that $w := [\![\mathcal{G}]\!]$ has length N, one can compute in time $\mathcal{O}(n)$ a data structure of size $\mathcal{O}(n)$ that allows to answer random access queries for w in time $\mathcal{O}(\log N)$.

- Very complicated proof, several sophisticated data structures!

The random access problem

Random access queries

For a string $w \in \Sigma^*$ of length N, a random access query gets a position $i \in [1, N]$ and returns the i-th symbol of w.

Theorem (Bille et al. 2015)

From a given SLP $\mathcal{G}$ of size n such that $w := [\![\mathcal{G}]\!]$ has length N, one can compute in time $\mathcal{O}(n)$ a data structure of size $\mathcal{O}(n)$ that allows to answer random access queries for w in time $\mathcal{O}(\log N)$.

- Very complicated proof, several sophisticated data structures!
- Size of the data structure is measures in number of words of bit length $\log N$.

The random access problem

Random access queries

For a string $w \in \Sigma^*$ of length N, a random access query gets a position $i \in [1, N]$ and returns the i-th symbol of w.

Theorem (Bille et al. 2015)

From a given SLP $\mathcal{G}$ of size n such that $w := [\![\mathcal{G}]\!]$ has length N, one can compute in time $\mathcal{O}(n)$ a data structure of size $\mathcal{O}(n)$ that allows to answer random access queries for w in time $\mathcal{O}(\log N)$.

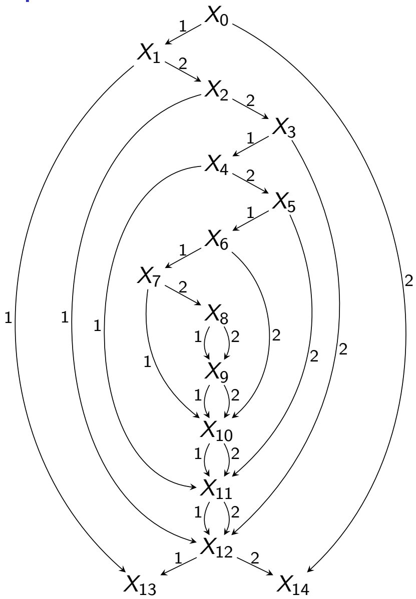
- Very complicated proof, several sophisticated data structures!
- Size of the data structure is measures in number of words of bit length $\log N$.
- Random access queries can be answered in time $\mathcal{O}(\text{depth}(\mathcal{G}))$ and space $\mathcal{O}(n)$.

Symmetric centroid path decomposition

$$
\begin{aligned}
X_0 &\to X_1 X_{14} \\
X_1 &\to X_{13} X_2 \\
X_2 &\to X_{12} X_3 \\
X_3 &\to X_4 X_{12} \\
X_4 &\to X_{11} X_5 \\
X_5 &\to X_6 X_{11} \\
X_6 &\to X_7 X_{10} \\
X_7 &\to X_{10} X_8 \\
X_8 &\to X_9 X_9 \\
X_9 &\to X_{10} X_{10} \\
X_{10} &\to X_{11} X_{11} \\
X_{11} &\to X_{12} X_{12} \\
X_{12} &\to X_{13} X_{14} \\
X_{13} &\to a \\
X_{14} &\to b
\end{aligned}
$$

Symmetric centroid path decomposition

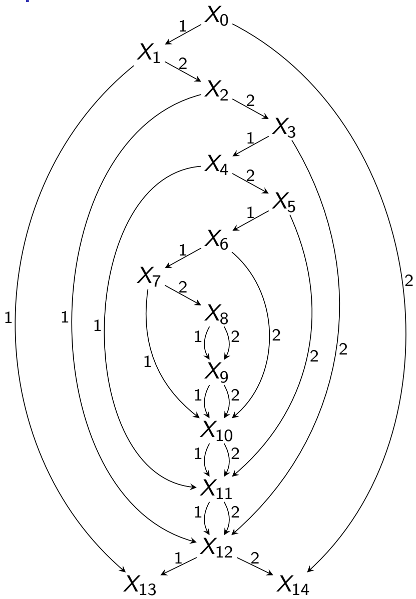
$$
\begin{aligned}
X_0 &\to X_1 X_{14} \\
X_1 &\to X_{13} X_2 \\
X_2 &\to X_{12} X_3 \\
X_3 &\to X_4 X_{12} \\
X_4 &\to X_{11} X_5 \\
X_5 &\to X_6 X_{11} \\
X_6 &\to X_7 X_{10} \\
X_7 &\to X_{10} X_8 \\
X_8 &\to X_9 X_9 \\
X_9 &\to X_{10} X_{10} \\
X_{10} &\to X_{11} X_{11} \\
X_{11} &\to X_{12} X_{12} \\
X_{12} &\to X_{13} X_{14} \\
X_{13} &\to a \\
X_{14} &\to b
\end{aligned}
$$



Symmetric centroid path decomposition

Label X_i with the pair consisting of

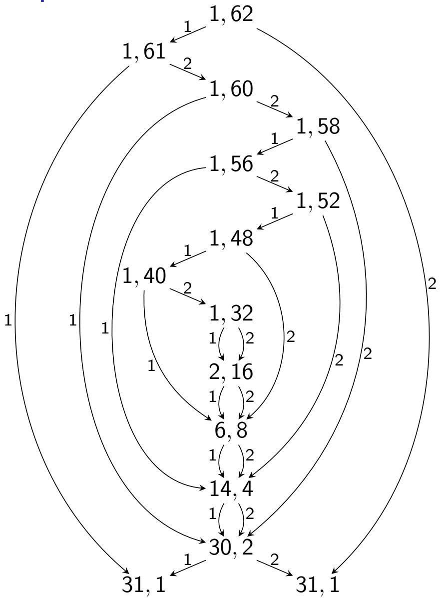
- the number of paths from the root X_0 to X_i, and
- the number of paths from X_i to the terminal variables X_{13} and X_{14}.



Symmetric centroid path decomposition

Label X_i with the pair consisting of

- the number of paths from the root X_0 to X_i, and
- the number of paths from X_i to the terminal variables X_{13} and X_{14}.

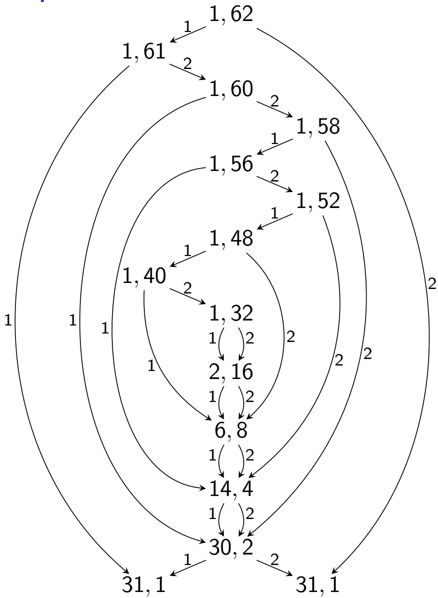


Symmetric centroid path decomposition

Label X_i with the pair consisting of

- the number of paths from the root X_0 to X_i, and
- the number of paths from X_i to the terminal variables X_{13} and X_{14}.

Take $\lfloor \log_2(\cdot) \rfloor$ for each number.

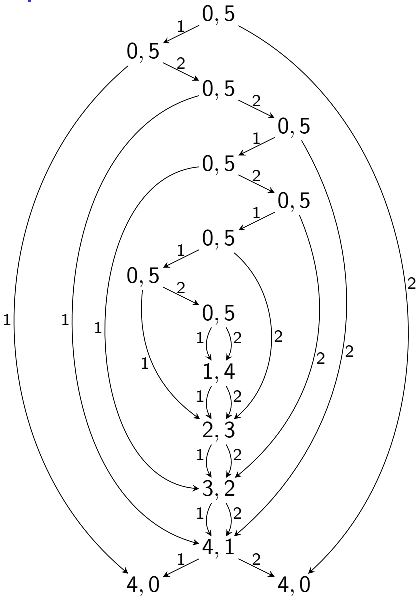


Symmetric centroid path decomposition

Label X_i with the pair consisting of

- the number of paths from the root X_0 to X_i, and
- the number of paths from X_i to the terminal variables X_{13} and X_{14}.

Take $\lfloor \log_2(\cdot) \rfloor$ for each number.



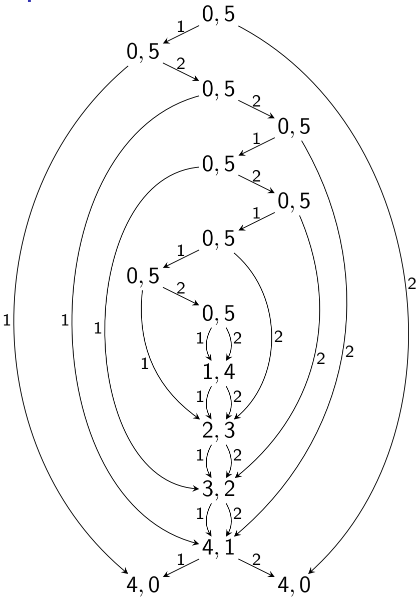
Symmetric centroid path decomposition

Label X_i with the pair consisting of

- the number of paths from the root X_0 to X_i, and
- the number of paths from X_i to the terminal variables X_{13} and X_{14}.

Take $\lfloor \log_2(\cdot) \rfloor$ for each number.

The symmetric centroid path decomposition consists of those edge connecting nodes that are labelled with the same pair.



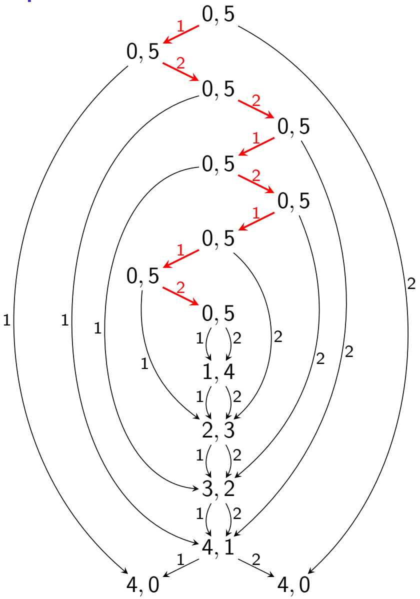
Symmetric centroid path decomposition

Label X_i with the pair consisting of

- the number of paths from the root X_0 to X_i, and
- the number of paths from X_i to the terminal variables X_{13} and X_{14}.

Take $\lfloor \log_2(\cdot) \rfloor$ for each number.

The symmetric centroid path decomposition consists of those edge connecting nodes that are labelled with the same pair.



Symmetric centroid path decomposition

Properties of the symmetric centroid path decomposition

- For every variable X_i at most one outgoing edge belongs to the symmetric centroid path decomposition.
- For every variable X_i at most one incoming edge belongs to the symmetric centroid path decomposition.
- Every path from the root X_0 to a terminal variable X_j contains a most $2 \cdot \log_2 N$ edges that do not belong to the symmetric centroid path decomposition. Here N is the length of the string $[\![\mathcal{G}]\!]$.

Symmetric centroid path decomposition

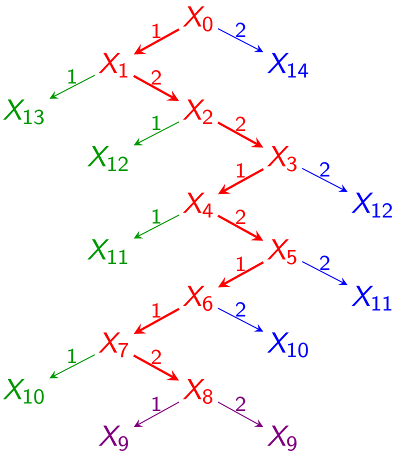
Properties of the symmetric centroid path decomposition

- For every variable X_i at most one outgoing edge belongs to the symmetric centroid path decomposition.
- For every variable X_i at most one incoming edge belongs to the symmetric centroid path decomposition.
- Every path from the root X_0 to a terminal variable X_j contains a most $2 \cdot \log_2 N$ edges that do not belong to the symmetric centroid path decomposition. Here N is the length of the string $[\![\mathcal{G}]\!]$.

By the first two properties the symmetric centroid path decomposition is indeed a decomposition of the DAG into paths (possibly of length zero).

Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part I

Consider a symmetric centroid path $X_s \to \cdots \to X_t$ (may consist only of X_t).

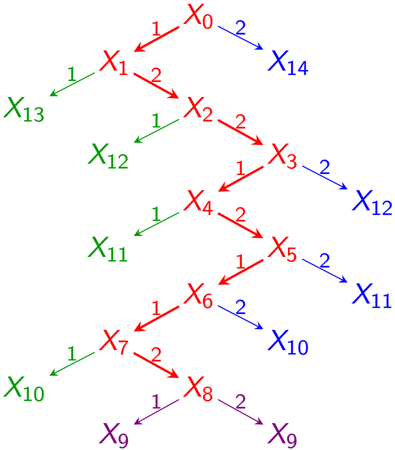


Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part I

Consider a symmetric centroid path $X_s \to \cdots \to X_t$ (may consist only of X_t).

For the last variable X_t on the path copy the $\mathcal{G}$-production:

- $X_8 \to X_9 X_9$.



Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part I

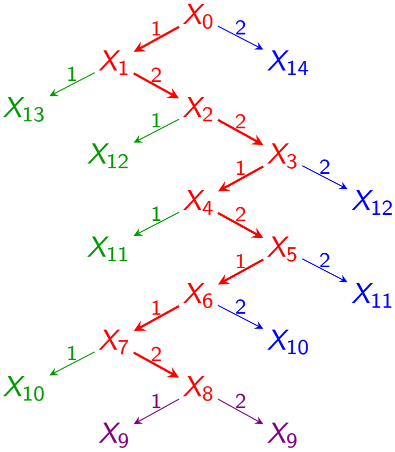
Consider a symmetric centroid path $X_s \to \cdots \to X_t$ (may consist only of X_t).

For the last variable X_t on the path copy the $\mathcal{G}$-production:

- $X_8 \to X_9 X_9$.

For all other variables X_i on the path add a new production $X_i \to L_i X_t R_i$:

- $X_i \to L_i X_8 R_i$ for $0 \leq i \leq 7$.



Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part I

Consider a symmetric centroid path $X_s \to \cdots \to X_t$ (may consist only of X_t).

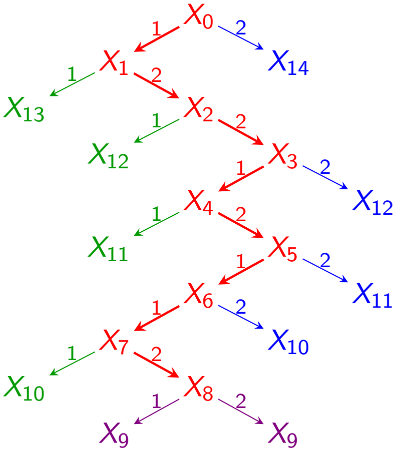
For the last variable X_t on the path copy the $\mathcal{G}$-production:

- $X_8 \to X_9 X_9$.

For all other variables X_i on the path add a new production $X_i \to L_i X_t R_i$:

- $X_i \to L_i X_8 R_i$ for $0 \le i \le 7$.

Add productions such that every L_i/R_i derives the appropriate suffix/prefix of $X_{13}X_{12}X_{11}X_{10}/X_{10}X_{11}X_{12}X_{14}$.



Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part II

The remaining problem

Given: word $y = Y_1 Y_2 \cdots Y_m$ of variables ($X_{13}X_{12}X_{11}X_{10}$ in our example)

Goal: construct an SLP $\mathcal{S}$ of size $\mathcal{O}(m)$ and “small depth” that contains for every suffix $s_i := Y_i Y_{i+1} \cdots Y_m$ a variable S_i that produces s_i.

Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part II

The remaining problem

Given: word $y = Y_1 Y_2 \cdots Y_m$ of variables ($X_{13} X_{12} X_{11} X_{10}$ in our example)

Goal: construct an SLP $\mathcal{S}$ of size $\mathcal{O}(m)$ and "small depth" that contains for every suffix $s_i := Y_i Y_{i+1} \cdots Y_m$ a variable S_i that produces s_i.

The corresponding problem for prefixes of y will be solved in the same way.

Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part II

The remaining problem

Given: word $y = Y_1 Y_2 \cdots Y_m$ of variables ($X_{13} X_{12} X_{11} X_{10}$ in our example)

Goal: construct an SLP $\mathcal{S}$ of size $\mathcal{O}(m)$ and "small depth" that contains for every suffix $s_i := Y_i Y_{i+1} \cdots Y_m$ a variable S_i that produces s_i.

The corresponding problem for prefixes of y will be solved in the same way.

What means "small depth"?

Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part II

The remaining problem

Given: word $y = Y_1 Y_2 \cdots Y_m$ of variables ($X_{13} X_{12} X_{11} X_{10}$ in our example)

Goal: construct an SLP $\mathcal{S}$ of size $\mathcal{O}(m)$ and "small depth" that contains for every suffix $s_i := Y_i Y_{i+1} \cdots Y_m$ a variable S_i that produces s_i.

The corresponding problem for prefixes of y will be solved in the same way.

What means "small depth"?

Assign to each variable Y_i the weight $\|Y_i\|$ = length of $[\![Y_i]\!]_{\mathcal{G}}$.

Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part II

The remaining problem

Given: word $y = Y_1 Y_2 \cdots Y_m$ of variables ($X_{13}X_{12}X_{11}X_{10}$ in our example)

Goal: construct an SLP $\mathcal{S}$ of size $\mathcal{O}(m)$ and "small depth" that contains for every suffix $s_i := Y_i Y_{i+1} \cdots Y_m$ a variable S_i that produces s_i.

The corresponding problem for prefixes of y will be solved in the same way.

What means "small depth"?

Assign to each variable Y_i the weight $\|Y_i\| =$ length of $[\![Y_i]\!]_{\mathcal{G}}$.

In our example: $\|X_{13}\| = 1$, $\|X_{12}\| = 2$, $\|X_{11}\| = 4$, $\|X_{10}\| = 8$.

Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part II

The remaining problem

Given: word $y = Y_1 Y_2 \cdots Y_m$ of variables ($X_{13}X_{12}X_{11}X_{10}$ in our example)

Goal: construct an SLP $\mathcal{S}$ of size $\mathcal{O}(m)$ and "small depth" that contains for every suffix $s_i := Y_i Y_{i+1} \cdots Y_m$ a variable S_i that produces s_i.

The corresponding problem for prefixes of y will be solved in the same way.

What means "small depth"?

Assign to each variable Y_i the weight $\|Y_i\| =$ length of $[\![Y_i]\!]_{\mathcal{G}}$.

In our example: $\|X_{13}\| = 1$, $\|X_{12}\| = 2$, $\|X_{11}\| = 4$, $\|X_{10}\| = 8$.

Extend the weight function $\|\cdot\|$ additively to words over the Y_i.

Construction of the balanced SLP $\mathcal{G}'$ from $\mathcal{G}$, Part II

The remaining problem

Given: word $y = Y_1 Y_2 \cdots Y_m$ of variables ($X_{13} X_{12} X_{11} X_{10}$ in our example)

Goal: construct an SLP $\mathcal{S}$ of size $\mathcal{O}(m)$ and "small depth" that contains for every suffix $s_i := Y_i Y_{i+1} \cdots Y_m$ a variable S_i that produces s_i.

The corresponding problem for prefixes of y will be solved in the same way.

What means "small depth"?

Assign to each variable Y_i the weight $\|Y_i\| =$ length of $[\![Y_i]\!]_{\mathcal{G}}$.

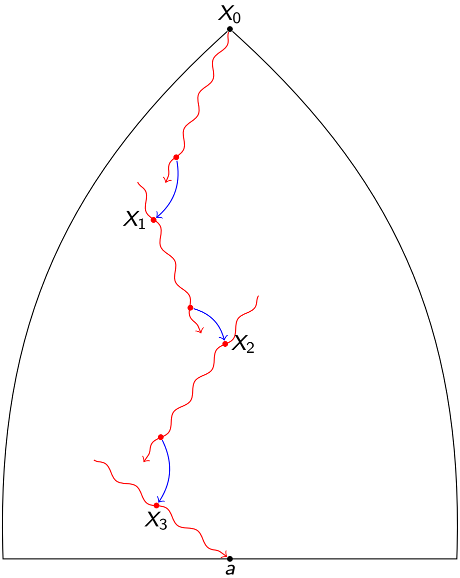
In our example: $\|X_{13}\| = 1$, $\|X_{12}\| = 2$, $\|X_{11}\| = 4$, $\|X_{10}\| = 8$.

Extend the weight function $\|\cdot\|$ additively to words over the Y_i.

Every path in the derivation tree of $\mathcal{S}$ from variable S_i to a terminal Y_j ($j \geq i$) has length $4 + 2 \log \|s_i\| - 2 \log_2 \|Y_j\|$.

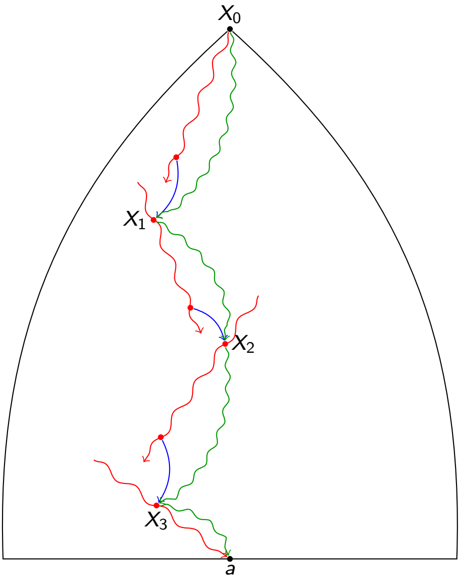
The global picture

- symmetric centroid paths
- edges not on symmetric centroid paths



The global picture

- symmetric centroid paths
- edges not on symmetric centroid paths
- a path in $\mathcal{G}'$



The global picture

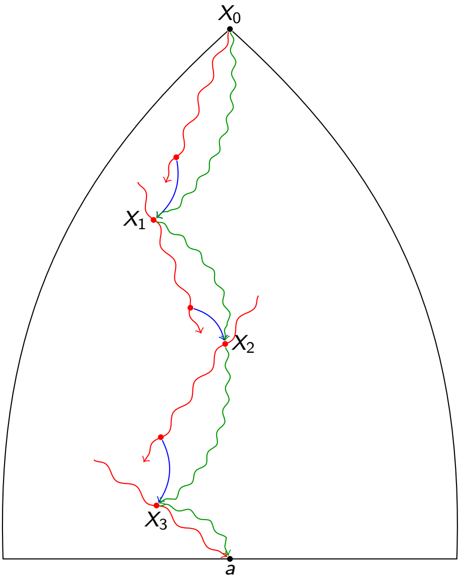
- symmetric centroid paths
- edges not on symmetric centroid paths
- a path in $\mathcal{G}'$
- lengths of green paths

  $5+2\log\|X_0\|-2\log_2\|X_1\|$

  $5+2\log\|X_1\|-2\log_2\|X_2\|$

  $5+2\log\|X_2\|-2\log_2\|X_3\|$

  $5+2\log\|X_3\|-2\log_2\|a\|$



The global picture

- symmetric centroid paths
- edges not on symmetric centroid paths
- a path in $\mathcal{G}'$
- lengths of green paths

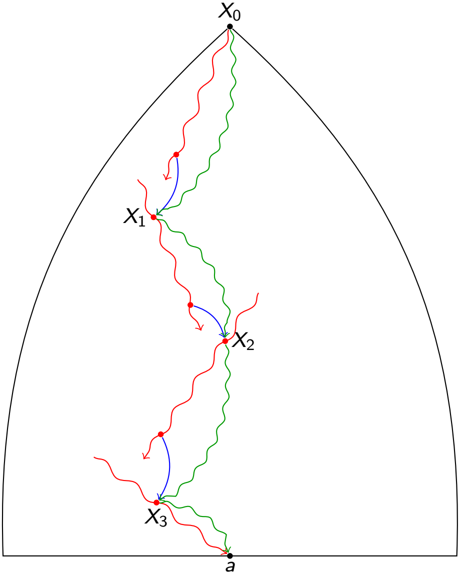
  $5+2\log\|X_0\|-2\log_2\|X_1\|$

  $5+2\log\|X_1\|-2\log_2\|X_2\|$

  $5+2\log\|X_2\|-2\log_2\|X_3\|$

  $5+2\log\|X_3\|-2\log_2\|a\|$

- total path length:
  $\leq \mathcal{O}(\log N) + 2\log\|X_0\|$
  $= \mathcal{O}(\log N)$



Suffixes of weighted strings: the main lemma

We consider strings over an alphabet Σ, where every $a \in \Sigma$ has a weight $\|a\|$, which is additively extended to strings over Σ.

Main lemma

For every weighted string s of length n one can construct in linear time an SLP $\mathcal{G}$ with the following properties:

- $\mathcal{G}$ contains at most $3n$ variables,
- all right-hand sides of $\mathcal{G}$ have length at most four,
- $\mathcal{G}$ contains variables $S_1, \ldots, S_n$ that produce the suffixes of s, and
- every path from a variable S_i to some terminal symbol a in the derivation tree has length $3 + 2(\log \|S_i\| - \log \|a\|)$.

Suffixes of weighted strings: proof of the main lemma

Recursive algorithm:

$n = 1$: obvious

Suffixes of weighted strings: proof of the main lemma

Recursive algorithm:

$n = 1$: obvious

$n \geq 2$: Let $s = ucv$ with $c \in \Sigma$, where cv is the shortest suffix of s such that $\lceil \log_2 \|cv\| \rceil = \lceil \log_2 \|s\| \rceil$.

Suffixes of weighted strings: proof of the main lemma

Recursive algorithm:

$n = 1$: obvious

$n \geq 2$: Let $s = ucv$ with $c \in \Sigma$, where cv is the shortest suffix of s such that $\lceil \log_2 \|cv\| \rceil = \lceil \log_2 \|s\| \rceil$.

- Recursive call for suffix v.
- Factorize u into block of length 2 and possibly one block of length 1.

  Introduce nonterminals $X_1, \ldots, X_k$ ($k = \lceil |u|/2 \rceil$) for the blocks.

  Recursive call for $X_1 X_2 \cdots X_k$, where the weight of X_i is the weight of the corresponding block.

Balancing tree grammars

SLPs have been extended to trees:

- tree straight-line programs (TSLPs)
- forest straight-line programs (FSLPs)
- top dags

Balancing tree grammars

SLPs have been extended to trees:

- tree straight-line programs (TSLPs)
- forest straight-line programs (FSLPs)
- top dags

Balancing TSLPs/FSLPs/top dags

From a given TSLP/FSLP/top dag $\mathcal{G}$ of size n producing a tree t of size N, one can construct in time $\mathcal{O}(n)$ a TSLP/FSLP/top dag $\mathcal{G}'$ such that:

- $\mathcal{G}'$ produces t as well,
- $|\mathcal{G}'| \in \mathcal{O}(n)$
- $\text{depth}(\mathcal{G}') \in \mathcal{O}(\log N)$

Balancing algebraic circuits: the meta theorem

Meta theorem for balancing circuits

Let $\mathcal{A}$ be an algebra with a certain finiteness property. From an algebraic circuit $\mathcal{G}$ over $\mathcal{A}$ one can compute an algebraic circuit $\mathcal{G}'$ such that:

- $\mathcal{G}'$ and $\mathcal{G}$ produces the same element of $\mathcal{A}$.
- $|\mathcal{G}'| \in \mathcal{O}(n)$
- $\text{depth}(\mathcal{G}') \in \mathcal{O}(\log N)$, where N is the size of the unravelling (unfolding) of $\mathcal{G}$.

An open problem

Is it possible to solve the random access problem for an SLP of size n producing a string of length N in

- time $\mathcal{O}(\log N / \log\log N)$
- using a data structure of size $\mathcal{O}(n)$?