On $\alpha\beta$-Factorization, Special Cases of Simon's Congruence, and Dominos

Jonas Höfer

Kiel University
GöKi-Workshop

September 14, 2023

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$$\texttt{aba} \preceq \texttt{aabbca} \qquad \texttt{acb} \not\preceq \texttt{aabbca}$$

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$\texttt{aba} \preceq \texttt{aabbca} \qquad \texttt{acb} \not\preceq \texttt{aabbca}$

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$\mathtt{aba} \preceq \mathtt{aabbca} \qquad \mathtt{acb} \not\preceq \mathtt{aabbca}$

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$\texttt{aba} \preceq \texttt{aabbca} \qquad \texttt{acb} \not\preceq \texttt{aabbca}$

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$$\texttt{aba} \preceq \texttt{aabbca} \qquad \texttt{acb} \not\preceq \texttt{aabbca}$$

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$\texttt{aba} \preceq \texttt{aabbca} \qquad \texttt{acb} \not\preceq \texttt{aabbca}$

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$\texttt{aba} \preceq \texttt{aabbca}$ $\texttt{acb} \not\preceq \texttt{aabbca}$

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$\texttt{aba} \preceq \texttt{aabbca}$ $\texttt{acb} \not\preceq \texttt{aabbca}$

- $u \preceq x$ and $v \preceq y$ implies $uv \preceq xy$

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$\mathtt{aba} \preceq \mathtt{aabbca}$ $\qquad$ $\mathtt{acb} \not\preceq \mathtt{aabbca}$

- $u \preceq x$ and $v \preceq y$ implies $uv \preceq xy$
- $uv \preceq w$ implies $w = xy$ s.t. $u \preceq x$ and $v \preceq y$

Introduction

Definition (Subword, Scattered Factor)

Let $v, w \in \Sigma^*$, then $v \preceq w \iff v$ can be embedded in w

$\texttt{aba} \preceq \texttt{aabbca}$ $\texttt{acb} \not\preceq \texttt{aabbca}$

- $u \preceq x$ and $v \preceq y$ implies $uv \preceq xy$
- $uv \preceq w$ implies $w = xy$ s.t. $u \preceq x$ and $v \preceq y$

Definition (Simon's Congruence)

For $u, v \in \Sigma^*$ we define $u \sim_k v \quad :\Longleftrightarrow \quad \text{ScatFact}_{\leq k}(u) = \text{ScatFact}_{\leq k}(v)$.

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

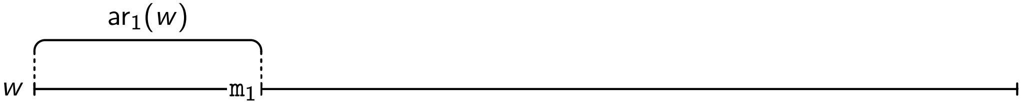
$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

[1]Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".
[2]Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

– Introduced by Fleischmann et al.[1] based on arch factorization[2]

[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

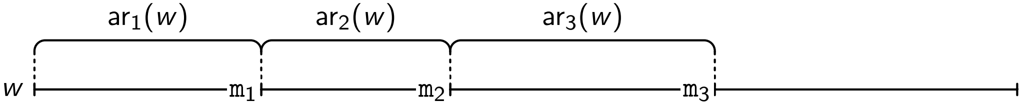
- Introduced by Fleischmann et al.[1] based on arch factorization[2]

[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".

[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

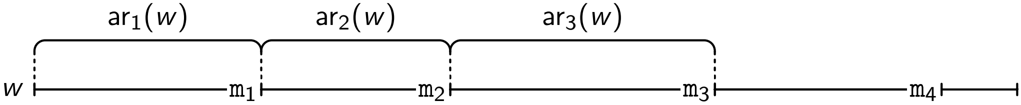


[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".

[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

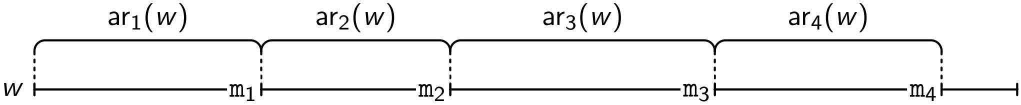
- Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

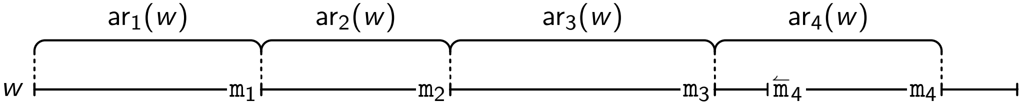
- Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., “Nearly *k*-Universal Words – Investigating a Part of Simon’s Congruence”.
[2] Hébrard, “An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences”.

$\alpha\beta$-factorization

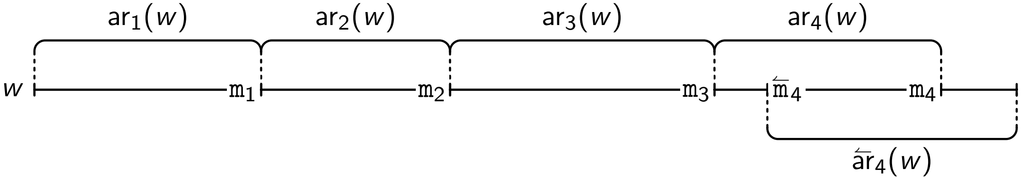
– Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

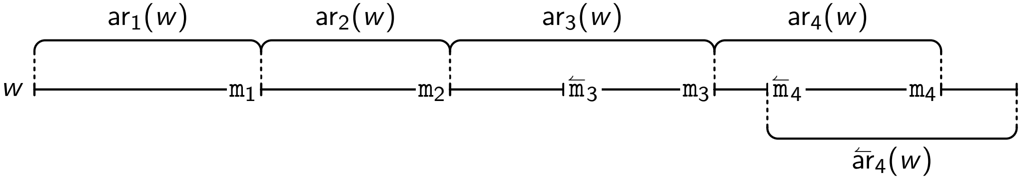
– Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

αβ-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

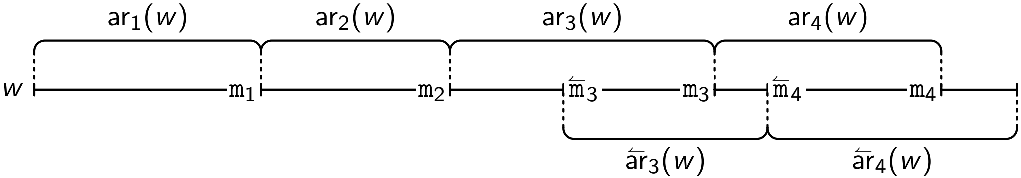


[1] Fleischmann et al., “Nearly *k*-Universal Words – Investigating a Part of Simon’s Congruence”.

[2] Hébrard, “An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences”.

$\alpha\beta$-factorization

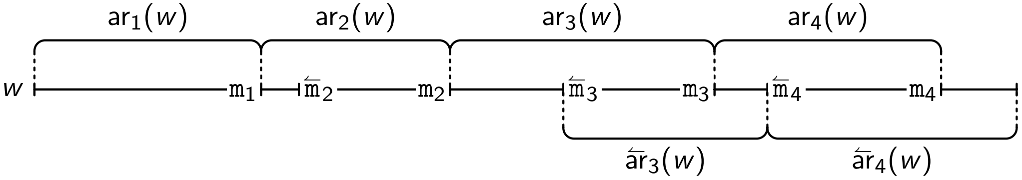
– Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

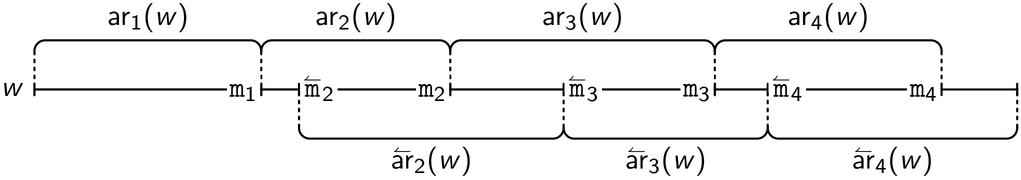
- Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

– Introduced by Fleischmann et al.[1] based on arch factorization[2]

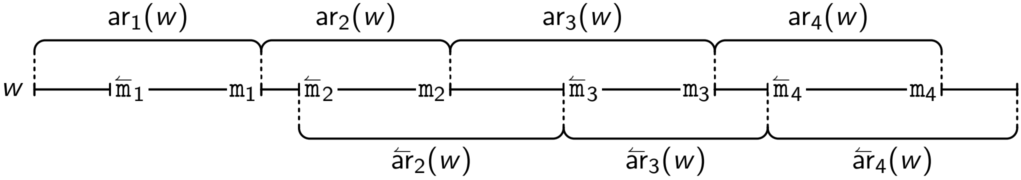


[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".

[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

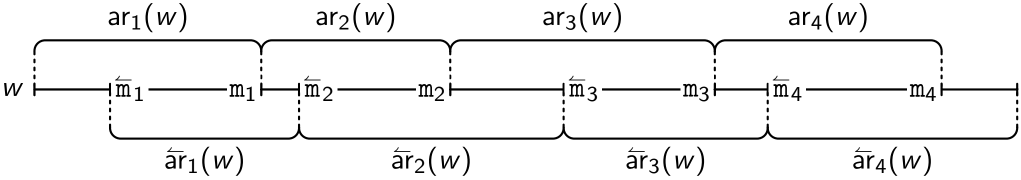
– Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

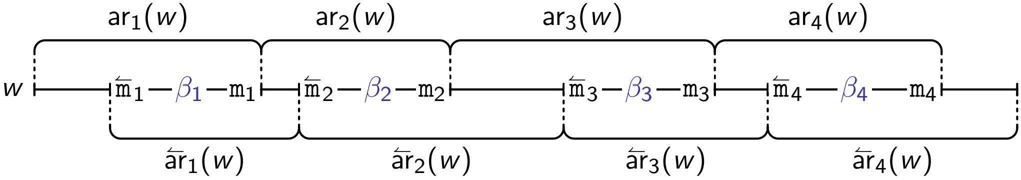
- Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

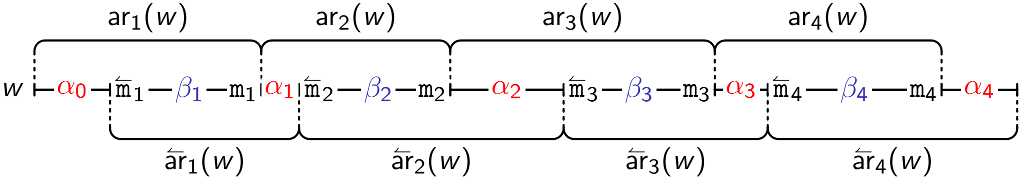


[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".

[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

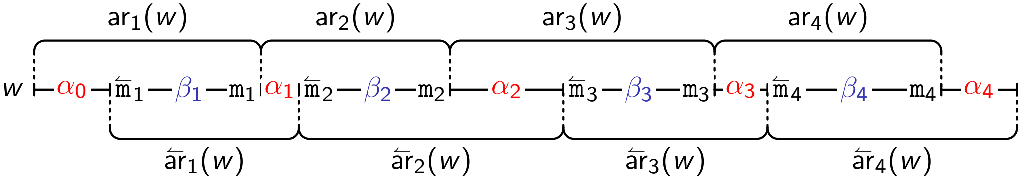
- Introduced by Fleischmann et al.[1] based on arch factorization[2]



[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

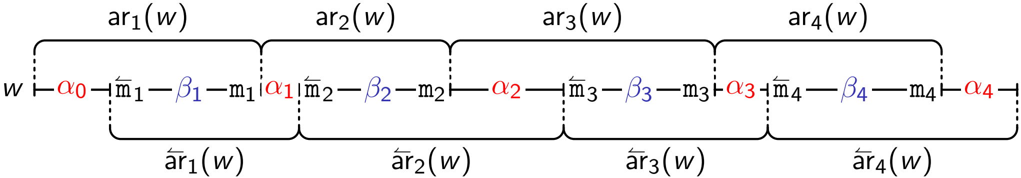


Example abaccabc

[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

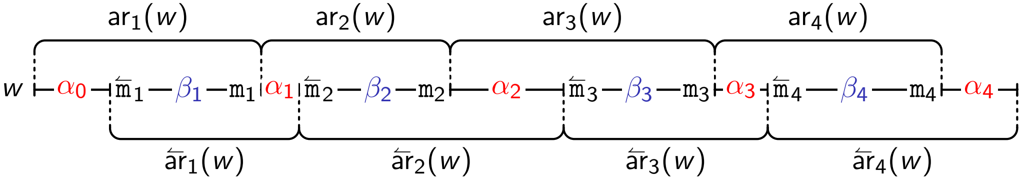


Example abac · cabc

[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

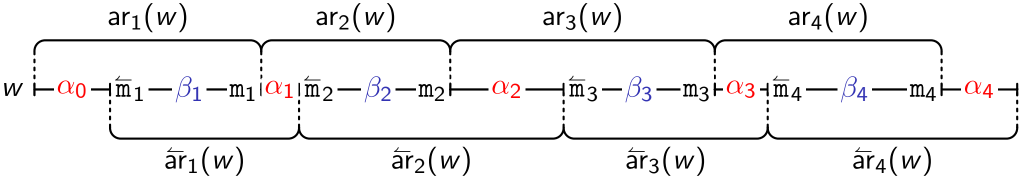


Example abac · cab · c

[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

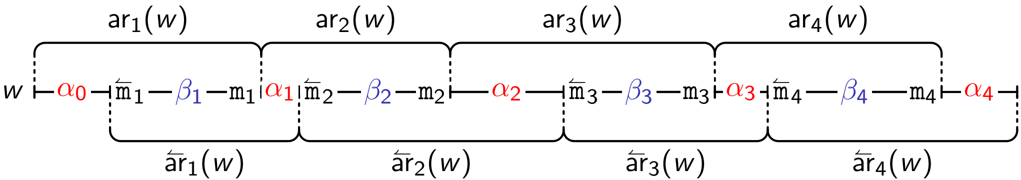


Example abac · c · ab · c

[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

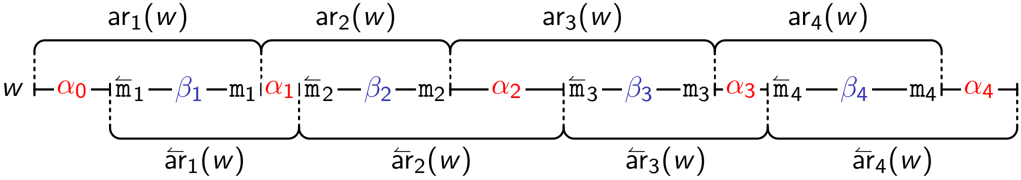


Example $\mathtt{a} \cdot \mathtt{bac} \cdot \mathtt{c} \cdot \mathtt{ab} \cdot \mathtt{c}$

[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

– Introduced by Fleischmann et al.[1] based on arch factorization[2]

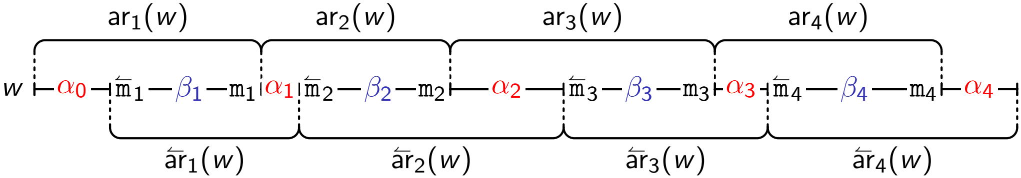


Example a · bac · c · ab · c

[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]

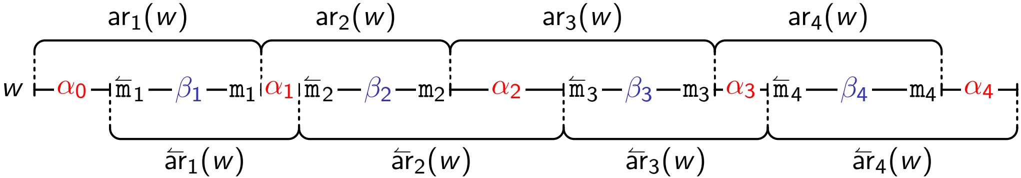


Example $\mathtt{a} \cdot \mathtt{bac} \cdot \mathtt{c} \cdot \mathtt{ab} \cdot \mathtt{c}$

[1] Fleischmann et al., "Nearly *k*-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

- Introduced by Fleischmann et al.[1] based on arch factorization[2]



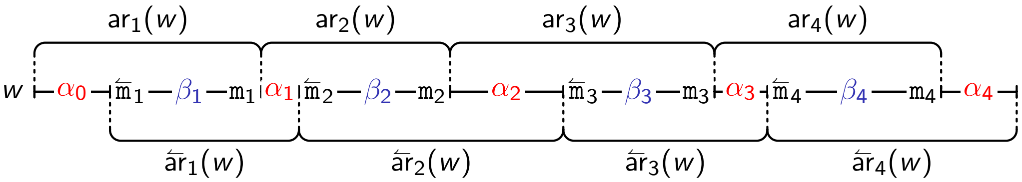
Example $\mathtt{a} \cdot \mathtt{bac} \cdot \mathtt{c} \cdot \mathtt{ab} \cdot \mathtt{c}$

Universality $\iota(w) := \#\text{arches} \iff \text{max. number s.t. } \Sigma^{\iota(w)} \subseteq \mathrm{ScatFact}(w)$

[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

$\alpha\beta$-factorization

– Introduced by Fleischmann et al.[1] based on arch factorization[2]



Example $\mathtt{a} \cdot \mathtt{bac} \cdot \mathtt{c} \cdot \mathtt{ab} \cdot \mathtt{c}$

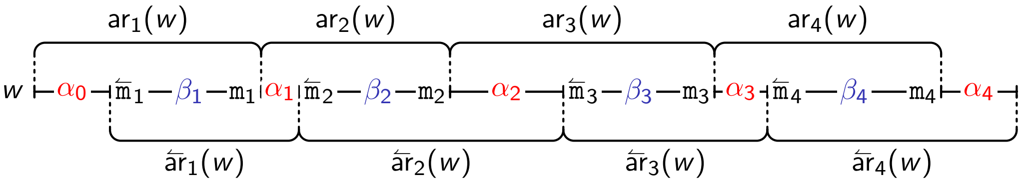
Universality $\iota(w) := \#\text{arches} \iff \text{max. number s.t. } \Sigma^{\iota(w)} \subseteq \text{ScatFact}(w)$

Facts – $\text{alph}(\text{ar}_i(w)) = \Sigma = \text{alph}(\overleftarrow{\text{ar}}_i(w))$ with first/last letter unique

[1] Fleischmann et al., “Nearly k-Universal Words – Investigating a Part of Simon’s Congruence”.
[2] Hébrard, “An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences”.

$\alpha\beta$-factorization

– Introduced by Fleischmann et al.[1] based on arch factorization[2]



Example $\texttt{a} \cdot \texttt{bac} \cdot \texttt{c} \cdot \texttt{ab} \cdot \texttt{c}$

Universality $\iota(w) := \#\text{arches} \iff \text{max. number s.t. } \Sigma^{\iota(w)} \subseteq \text{ScatFact}(w)$

Facts
– $\text{alph}(\text{ar}_i(w)) = \Sigma = \text{alph}(\overleftarrow{\text{ar}}_i(w))$ with first/last letter unique
– $\text{suff}_1(\beta_i) = \text{m}_i(w) \not\preceq \alpha_{i-1}$ and $\text{pref}_1(\beta_i) = \overleftarrow{\text{m}}_i(w) \not\preceq \alpha_i$

[1] Fleischmann et al., "Nearly k-Universal Words – Investigating a Part of Simon's Congruence".
[2] Hébrard, "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences".

General Results on $\alpha\beta$-Factorization

Lemma

$u = \alpha_0\beta_1\cdots\alpha_m, v = \alpha'_0\beta'_1\cdots\alpha'_m \in \Sigma^*$ *with* $u \sim_k v$ *and* $m := \iota(u) = \iota(v) < k$, *then*

- $\alpha_i\beta_{i+1}\alpha_{i+1}\cdots\beta_j\alpha_j \sim_{k-m+(j-i)} \alpha'_i\beta'_{i+1}\alpha'_{i+1}\cdots\beta'_j\alpha'_j$ *for* $0 \leq i \leq j \leq m$ *we have*
- *specifically,* $\alpha_i \sim_{k-m} \alpha'_i$ *for* $i \in [m]_0$

General Results on $\alpha\beta$-Factorization

Lemma

$u = \alpha_0\beta_1 \cdots \alpha_m, v = \alpha'_0\beta'_1 \cdots \alpha'_m \in \Sigma^*$ *with* $u \sim_k v$ *and* $m := \iota(u) = \iota(v) < k$, *then*

- $\alpha_i\beta_{i+1}\alpha_{i+1} \cdots \beta_j\alpha_j \sim_{k-m+(j-i)} \alpha'_i\beta'_{i+1}\alpha'_{i+1} \cdots \beta'_j\alpha'_j$ *for* $0 \le i \le j \le m$ *we have*
- *specifically,* $\alpha_i \sim_{k-m} \alpha'_i$ *for* $i \in [m]_0$

Proof Sketch. we show $u \sim_k v \implies \mathrm{ar}_1^{-1}(u) \cdot u \sim_{k-1} \mathrm{ar}_1^{-1}(v) \cdot v$, then induction

$\vdash \mathrm{ar}_1(u) \dashv\vdash \tilde{u} := \mathrm{ar}_1^{-1}(u) \cdot u \dashv \quad u \qquad v \quad \vdash \mathrm{ar}_1(v)\text{-}\mathtt{m}_1 \dashv\vdash \tilde{v} := \mathrm{ar}_1^{-1}(v) \cdot v \dashv$

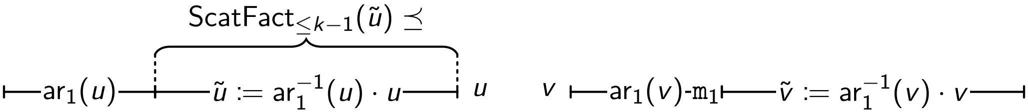
General Results on $\alpha\beta$-Factorization

Lemma

$u = \alpha_0\beta_1 \cdots \alpha_m, v = \alpha'_0\beta'_1 \cdots \alpha'_m \in \Sigma^*$ *with* $u \sim_k v$ *and* $m := \iota(u) = \iota(v) < k$*, then*

- $\alpha_i\beta_{i+1}\alpha_{i+1} \cdots \beta_j\alpha_j \sim_{k-m+(j-i)} \alpha'_i\beta'_{i+1}\alpha'_{i+1} \cdots \beta'_j\alpha'_j$ *for* $0 \leq i \leq j \leq m$ *we have*
- *specifically,* $\alpha_i \sim_{k-m} \alpha'_i$ *for* $i \in [m]_0$

Proof Sketch. we show $u \sim_k v \implies \mathrm{ar}_1^{-1}(u) \cdot u \sim_{k-1} \mathrm{ar}_1^{-1}(v) \cdot v$, then induction



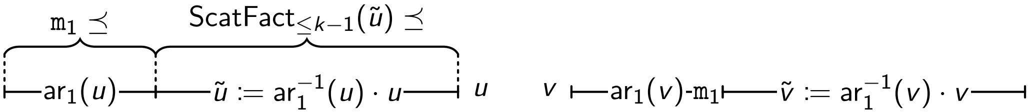
General Results on $\alpha\beta$-Factorization

Lemma

$u = \alpha_0\beta_1 \cdots \alpha_m, v = \alpha'_0\beta'_1 \cdots \alpha'_m \in \Sigma^*$ with $u \sim_k v$ and $m := \iota(u) = \iota(v) < k$, then

- *$\alpha_i\beta_{i+1}\alpha_{i+1} \cdots \beta_j\alpha_j \sim_{k-m+(j-i)} \alpha'_i\beta'_{i+1}\alpha'_{i+1} \cdots \beta'_j\alpha'_j$ for $0 \leq i \leq j \leq m$ we have*
- *specifically, $\alpha_i \sim_{k-m} \alpha'_i$ for $i \in [m]_0$*

Proof Sketch. we show $u \sim_k v \implies \mathrm{ar}_1^{-1}(u) \cdot u \sim_{k-1} \mathrm{ar}_1^{-1}(v) \cdot v$, then induction



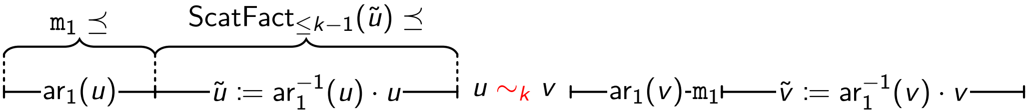
General Results on $\alpha\beta$-Factorization

Lemma

$u = \alpha_0\beta_1\cdots\alpha_m, v = \alpha'_0\beta'_1\cdots\alpha'_m \in \Sigma^*$ with $u \sim_k v$ and $m := \iota(u) = \iota(v) < k$, then

- *$\alpha_i\beta_{i+1}\alpha_{i+1}\cdots\beta_j\alpha_j \sim_{k-m+(j-i)} \alpha'_i\beta'_{i+1}\alpha'_{i+1}\cdots\beta'_j\alpha'_j$ for $0 \le i \le j \le m$ we have*
- *specifically, $\alpha_i \sim_{k-m} \alpha'_i$ for $i \in [m]_0$*

Proof Sketch. we show $u \sim_k v \implies \mathrm{ar}_1^{-1}(u)\cdot u \sim_{k-1} \mathrm{ar}_1^{-1}(v)\cdot v$, then induction



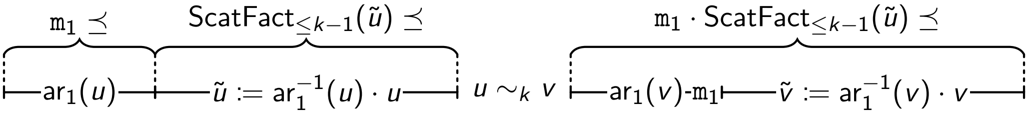
General Results on $\alpha\beta$-Factorization

Lemma

$u = \alpha_0\beta_1\cdots\alpha_m, v = \alpha'_0\beta'_1\cdots\alpha'_m \in \Sigma^*$ with $u \sim_k v$ and $m := \iota(u) = \iota(v) < k$, then

- *$\alpha_i\beta_{i+1}\alpha_{i+1}\cdots\beta_j\alpha_j \sim_{k-m+(j-i)} \alpha'_i\beta'_{i+1}\alpha'_{i+1}\cdots\beta'_j\alpha'_j$ for $0 \le i \le j \le m$ we have*
- *specifically, $\alpha_i \sim_{k-m} \alpha'_i$ for $i \in [m]_0$*

Proof Sketch. we show $u \sim_k v \implies \mathrm{ar}_1^{-1}(u)\cdot u \sim_{k-1} \mathrm{ar}_1^{-1}(v)\cdot v$, then induction



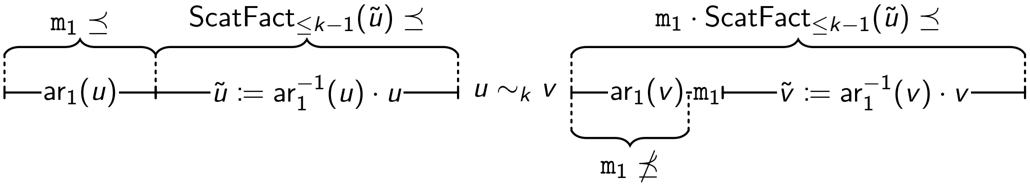
General Results on $\alpha\beta$-Factorization

Lemma

$u = \alpha_0\beta_1 \cdots \alpha_m, v = \alpha'_0\beta'_1 \cdots \alpha'_m \in \Sigma^*$ with $u \sim_k v$ and $m := \iota(u) = \iota(v) < k$, then

- *$\alpha_i\beta_{i+1}\alpha_{i+1} \cdots \beta_j\alpha_j \sim_{k-m+(j-i)} \alpha'_i\beta'_{i+1}\alpha'_{i+1} \cdots \beta'_j\alpha'_j$ for $0 \leq i \leq j \leq m$ we have*
- *specifically, $\alpha_i \sim_{k-m} \alpha'_i$ for $i \in [m]_0$*

Proof Sketch. we show $u \sim_k v \implies \mathrm{ar}_1^{-1}(u) \cdot u \sim_{k-1} \mathrm{ar}_1^{-1}(v) \cdot v$, then induction



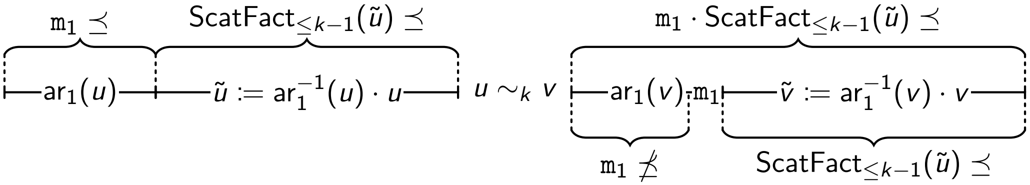
General Results on $\alpha\beta$-Factorization

Lemma

$u = \alpha_0\beta_1\cdots\alpha_m, v = \alpha'_0\beta'_1\cdots\alpha'_m \in \Sigma^*$ with $u \sim_k v$ and $m := \iota(u) = \iota(v) < k$, then

- *$\alpha_i\beta_{i+1}\alpha_{i+1}\cdots\beta_j\alpha_j \sim_{k-m+(j-i)} \alpha'_i\beta'_{i+1}\alpha'_{i+1}\cdots\beta'_j\alpha'_j$ for $0 \leq i \leq j \leq m$ we have*
- *specifically, $\alpha_i \sim_{k-m} \alpha'_i$ for $i \in [m]_0$*

Proof Sketch. we show $u \sim_k v \implies \mathrm{ar}_1^{-1}(u)\cdot u \sim_{k-1} \mathrm{ar}_1^{-1}(v)\cdot v$, then induction



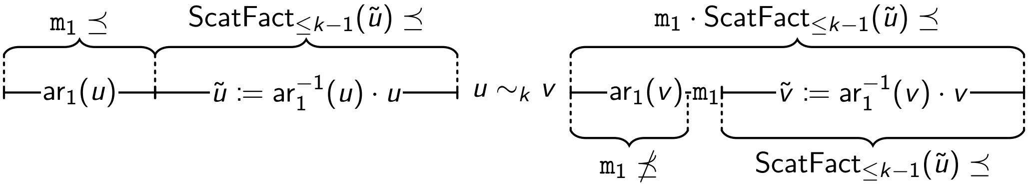
General Results on $\alpha\beta$-Factorization

Lemma

$u = \alpha_0\beta_1\cdots\alpha_m, v = \alpha'_0\beta'_1\cdots\alpha'_m \in \Sigma^*$ *with* $u \sim_k v$ *and* $m := \iota(u) = \iota(v) < k$, *then*

- $\alpha_i\beta_{i+1}\alpha_{i+1}\cdots\beta_j\alpha_j \sim_{k-m+(j-i)} \alpha'_i\beta'_{i+1}\alpha'_{i+1}\cdots\beta'_j\alpha'_j$ *for* $0 \leq i \leq j \leq m$ *we have*
- *specifically,* $\alpha_i \sim_{k-m} \alpha'_i$ *for* $i \in [m]_0$

Proof Sketch. we show $u \sim_k v \implies \mathrm{ar}_1^{-1}(u)\cdot u \sim_{k-1} \mathrm{ar}_1^{-1}(v)\cdot v$, then induction



thus $\mathrm{ScatFact}_{\leq k-1}(\tilde{u}) \subseteq \mathrm{ScatFact}_{\leq k-1}(\tilde{v})$ and "$\supseteq$" by symmetry □

General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_\ell \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max \begin{pmatrix} \min(k_1 + \iota(v), \iota(\tilde{u}) + k_2) \\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2) \end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_{\ell} \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max \begin{pmatrix} \min(k_1 + \iota(v), \iota(\tilde{u}) + k_2) \\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2) \end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

Proof Sketch. Following Karandikar et al. we prove $x \sim_k y$ implies $zx \sim_{\iota(z)+k} zy$.

[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

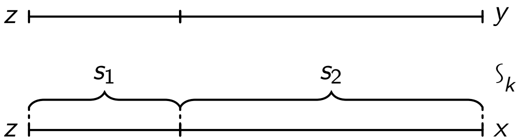
General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_\ell \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max \begin{pmatrix} \min(k_1 + \iota(v), \iota(\tilde{u}) + k_2) \\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2) \end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

Proof Sketch. Following Karandikar et al. we prove $x \sim_k y$ implies $zx \sim_{\iota(z)+k} zy$. Consider *left-most* occurence of $s \preceq zx$. Thus, $s_1 \cdot s_2 \preceq z \cdot x$ with $\iota(z) \leq |s_1|$.



[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

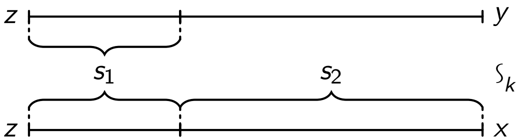
General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_\ell \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max \begin{pmatrix} \min(k_1 + \iota(v), \iota(\tilde{u}) + k_2) \\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2) \end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

Proof Sketch. Following Karandikar et al. we prove $x \sim_k y$ implies $zx \sim_{\iota(z)+k} zy$. Consider *left-most* occurence of $s \preceq zx$. Thus, $s_1 \cdot s_2 \preceq z \cdot x$ with $\iota(z) \leq |s_1|$.



[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

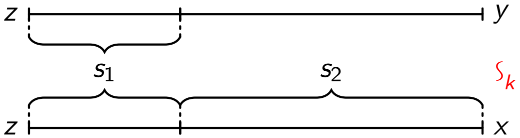
General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_{\ell} \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max\begin{pmatrix}\min(k_1 + \iota(v), \iota(\tilde{u}) + k_2) \\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2)\end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

Proof Sketch. Following Karandikar et al. we prove $x \sim_k y$ implies $zx \sim_{\iota(z)+k} zy$. Consider *left-most* occurence of $s \preceq zx$. Thus, $s_1 \cdot s_2 \preceq z \cdot x$ with $\iota(z) \leq |s_1|$.



[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

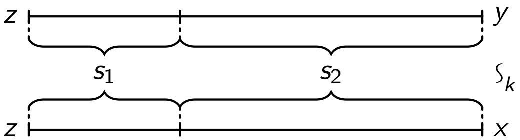
General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_\ell \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max \begin{pmatrix} \min(k_1 + \iota(v), \iota(\tilde{u}) + k_2) \\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2) \end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

Proof Sketch. Following Karandikar et al. we prove $x \sim_k y$ implies $zx \sim_{\iota(z)+k} zy$. Consider *left-most* occurence of $s \preceq zx$. Thus, $s_1 \cdot s_2 \preceq z \cdot x$ with $\iota(z) \leq |s_1|$.



[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_{\ell} \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max\begin{pmatrix}\min(k_1 + \iota(v), \iota(\tilde{u}) + k_2)\\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2)\end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

Proof Sketch. Using, $x \sim_k y$ implies $zx \sim_{k+\iota(z)} zy$, we conclude the claim.

[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_\ell \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max \begin{pmatrix} \min(k_1 + \iota(v), \iota(\tilde{u}) + k_2) \\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2) \end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

Proof Sketch. Using, $x \sim_k y$ implies $zx \sim_{k+\iota(z)} zy$, we conclude the claim. Consider the following square.

$$\begin{array}{ccc} u\tilde{v} & \sim_{k_1+\iota(v)} & \tilde{u}\tilde{v} \\ \wr_{\iota(\tilde{u})+k_2} & & \wr_{\iota(u)+k_2} \\ uv & \sim_{k_1+\iota(\tilde{v})} & \tilde{u}v \end{array}$$

[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_{\ell} \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max \begin{pmatrix} \min(k_1 + \iota(v), \iota(\tilde{u}) + k_2) \\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2) \end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

Proof Sketch. Using, $x \sim_k y$ implies $zx \sim_{k+\iota(z)} zy$, we conclude the claim. Consider the following square.

$$\begin{array}{ccc} u\tilde{v} & \sim_{k_1+\iota(v)} & \tilde{u}\tilde{v} \\ \wr_{\iota(\tilde{u})+k_2} & & \wr_{\iota(u)+k_2} \\ uv & \sim_{k_1+\iota(\tilde{v})} & \tilde{u}v \end{array}$$

[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

General Results on $\alpha\beta$-Factorization

Lemma (Based on Lemma by Karandikar et al.[3])

Let $u \sim_{k_1} \tilde{u}, v \sim_{k_2} \tilde{v}$ *then* $uv \sim_\ell \tilde{u}\tilde{v}$ *with* $m_1 := \min(\iota(u), \iota(\tilde{u})), m_2 := \min(\iota(v), \iota(\tilde{v}))$

$$\ell := \max \begin{pmatrix} \min(k_1 + \iota(v), \iota(\tilde{u}) + k_2) \\ \min(k_1 + \iota(\tilde{v}), \iota(u) + k_2) \end{pmatrix} \geq \min(k_1 + m_2, k_2 + m_1)$$

Proof Sketch. Using, $x \sim_k y$ implies $zx \sim_{k+\iota(z)} zy$, we conclude the claim. Consider the following square.

$$\begin{array}{ccc} u\tilde{v} & \sim_{k_1+\iota(v)} & \tilde{u}\tilde{v} \\ \wr_{\iota(\tilde{u})+k_2} & & \wr_{\iota(u)+k_2} \\ uv & \sim_{k_1+\iota(\tilde{v})} & \tilde{u}v \end{array}$$

[3] Karandikar et al., "On the Index of Simon's Congruence for Piecewise Testability".

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ & \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ & \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0.\alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$u = \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m$$
$$\tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\quad\; \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ & \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\quad\ \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\quad\ \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2 \cdots \beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2 \cdots \beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\alpha_2 \cdots \beta_{m-1}\alpha_m \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\dots \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\dots \\ &\sim_k \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\tilde{\alpha}_2\cdots\beta_{m-1}\tilde{\alpha}_m \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\cdots \\ &\sim_k \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\tilde{\alpha}_2\cdots\beta_{m-1}\tilde{\alpha}_m \\ &= \tilde{u} \end{aligned}$$

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Proof Sketch. We use $x \sim_\ell y \implies xz \sim_{\ell+\iota(z)} yz$ (move at least $\iota(z)$ letters into z):

$$\begin{aligned} u &= \alpha_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\alpha_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\sim_k \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\alpha_2\cdots\beta_{m-1}\alpha_m \\ &\cdots \\ &\sim_k \tilde{\alpha}_0\beta_1\tilde{\alpha}_1\beta_2\tilde{\alpha}_2\cdots\beta_{m-1}\tilde{\alpha}_m \\ &= \tilde{u} \end{aligned}$$

□

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\beta_i = \tilde{\beta}_i$ for all $i \in [m]$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Corollary

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, and $\hat{u} := \alpha_0 \tilde{\beta}_1 \alpha_1 \cdots \tilde{\beta}_m \alpha_m$, then

$$u \sim_k \tilde{u} \iff u \sim_k \hat{u}.$$

– W.l.o.g. we can exchange the α factors to be equal

General Results on $\alpha\beta$-Factorization

Theorem

$u, \tilde{u} \in \Sigma^*$ *with* $m := \iota(u) = \iota(\tilde{u}) < k$, *and* $\beta_i = \tilde{\beta}_i$ *for all* $i \in [m]$, *then*

$$u \sim_k \tilde{u} \iff \forall i \in [m]_0 . \alpha_i \sim_{k-m} \tilde{\alpha}_i$$

Corollary

$u, \tilde{u} \in \Sigma^*$ *with* $m := \iota(u) = \iota(\tilde{u}) < k$, *and* $\hat{u} := \alpha_0 \tilde{\beta}_1 \alpha_1 \cdots \tilde{\beta}_m \alpha_m$, *then*

$$u \sim_k \tilde{u} \iff u \sim_k \hat{u}.$$

– W.l.o.g. we can exchange the α factors to be equal

General Results on $\alpha\beta$-Factorization

A more general decomposition theorem

Theorem ($\alpha\beta\alpha$-decomposition)

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m].\alpha_{i-1}\beta_i\alpha_i \sim_{k-m+1} \tilde{\alpha}_{i-1}\tilde{\beta}_i\tilde{\alpha}_i$$

General Results on $\alpha\beta$-Factorization

A more general decomposition theorem

Theorem ($\alpha\beta\alpha$-decomposition)

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m].\alpha_{i-1}\beta_i\alpha_i \sim_{k-m+1} \tilde{\alpha}_{i-1}\tilde{\beta}_i\tilde{\alpha}_i$$

Proof Sketch. We apply the previous corollary. Then we use an exchange argument.

$$u \sim_k \alpha_0\tilde{\beta}_1\alpha_1\beta_2\alpha_2\cdots\beta_m\alpha_m \sim_k \alpha_0\tilde{\beta}_1\alpha_1\tilde{\beta}_2\alpha_2\cdots\beta_m\alpha_m \sim_k \ldots \sim_k \hat{u}.$$

General Results on $\alpha\beta$-Factorization

A more general decomposition theorem

Theorem ($\alpha\beta\alpha$-decomposition)

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m].\alpha_{i-1}\beta_i\alpha_i \sim_{k-m+1} \tilde{\alpha}_{i-1}\tilde{\beta}_i\tilde{\alpha}_i$$

Proof Sketch. We apply the previous corollary. Then we use an exchange argument.

$$u \sim_k \alpha_0\tilde{\beta}_1\alpha_1\beta_2\alpha_2 \cdots \beta_m\alpha_m \sim_k \alpha_0\tilde{\beta}_1\alpha_1\tilde{\beta}_2\alpha_2 \cdots \beta_m\alpha_m \sim_k \ldots \sim_k \hat{u}.$$

General Results on $\alpha\beta$-Factorization

A more general decomposition theorem

Theorem ($\alpha\beta\alpha$-decomposition)

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m].\alpha_{i-1}\beta_i\alpha_i \sim_{k-m+1} \tilde{\alpha}_{i-1}\tilde{\beta}_i\tilde{\alpha}_i$$

Proof Sketch. We apply the previous corollary. Then we use an exchange argument.

$$u \sim_k \alpha_0\tilde{\beta}_1\alpha_1\beta_2\alpha_2 \cdots \beta_m\alpha_m \sim_k \alpha_0\tilde{\beta}_1\alpha_1\tilde{\beta}_2\alpha_2 \cdots \beta_m\alpha_m \sim_k \ldots \sim_k \hat{u}.$$

General Results on $\alpha\beta$-Factorization

A more general decomposition theorem

Theorem ($\alpha\beta\alpha$-decomposition)

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m].\alpha_{i-1}\beta_i\alpha_i \sim_{k-m+1} \tilde{\alpha}_{i-1}\tilde{\beta}_i\tilde{\alpha}_i$$

Proof Sketch. We apply the previous corollary. Then we use an exchange argument.

$$u \sim_k \alpha_0\tilde{\beta}_1\alpha_1\beta_2\alpha_2\cdots\beta_m\alpha_m \sim_k \alpha_0\tilde{\beta}_1\alpha_1\tilde{\beta}_2\alpha_2\cdots\beta_m\alpha_m \sim_k \ldots \sim_k \hat{u}.$$

□

General Results on $\alpha\beta$-Factorization

A more general decomposition theorem

Theorem ($\alpha\beta\alpha$-decomposition)

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m].\alpha_{i-1}\beta_i\alpha_i \sim_{k-m+1} \tilde{\alpha}_{i-1}\tilde{\beta}_i\tilde{\alpha}_i$$

Proof Sketch. We apply the previous corollary. Then we use an exchange argument.

$$u \sim_k \alpha_0\tilde{\beta}_1\alpha_1\beta_2\alpha_2\cdots\beta_m\alpha_m \sim_k \alpha_0\tilde{\beta}_1\alpha_1\tilde{\beta}_2\alpha_2\cdots\beta_m\alpha_m \sim_k \ldots \sim_k \hat{u}.$$ □

– reduces problem to 1-universal words

General Results on $\alpha\beta$-Factorization

A more general decomposition theorem

Theorem ($\alpha\beta\alpha$-decomposition)

$u, \tilde{u} \in \Sigma^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m].\alpha_{i-1}\beta_i\alpha_i \sim_{k-m+1} \tilde{\alpha}_{i-1}\tilde{\beta}_i\tilde{\alpha}_i$$

Proof Sketch. We apply the previous corollary. Then we use an exchange argument.

$$u \sim_k \alpha_0\tilde{\beta}_1\alpha_1\beta_2\alpha_2\cdots\beta_m\alpha_m \sim_k \alpha_0\tilde{\beta}_1\alpha_1\tilde{\beta}_2\alpha_2\cdots\beta_m\alpha_m \sim_k \ldots \sim_k \hat{u}.$$

□

– reduces problem to 1-universal words
– analysis of 1-universal words is hard in general

The Binary Case

- $\beta_i \in \{\mathtt{ab}, \mathtt{ba}, \mathtt{a}, \mathtt{b}\}$ by uniqueness of first and last letter
- $\alpha_i \in \mathtt{a}^* \cup \mathtt{b}^*$ since $\mathrm{alph}(\alpha_i) \subset \Sigma_2$

The Binary Case

- $\beta_i \in \{\mathtt{ab}, \mathtt{ba}, \mathtt{a}, \mathtt{b}\}$ by uniqueness of first and last letter
- $\alpha_i \in \mathtt{a}^* \cup \mathtt{b}^*$ since $\mathrm{alph}(\alpha_i) \subset \Sigma_2$

Lemma

$u, \tilde{u} \in \Sigma_2^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$ and $u \sim_k \tilde{u}$, then $\beta_i = \tilde{\beta}_i$

The Binary Case

- $\beta_i \in \{\mathtt{ab}, \mathtt{ba}, \mathtt{a}, \mathtt{b}\}$ by uniqueness of first and last letter
- $\alpha_i \in \mathtt{a}^* \cup \mathtt{b}^*$ since $\mathsf{alph}(\alpha_i) \subset \Sigma_2$

Lemma

$u, \tilde{u} \in \Sigma_2^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$ and $u \sim_k \tilde{u}$, then $\beta_i = \tilde{\beta}_i$

- Follows from case distinctions on $\alpha\beta\alpha$-triples
- Equivalence is completely determined by pairwise equivalence of factors

The Binary Case

- $\beta_i \in \{\mathtt{ab}, \mathtt{ba}, \mathtt{a}, \mathtt{b}\}$ by uniqueness of first and last letter
- $\alpha_i \in \mathtt{a}^* \cup \mathtt{b}^*$ since $\mathrm{alph}(\alpha_i) \subset \Sigma_2$

Lemma

$u, \tilde{u} \in \Sigma_2^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$ and $u \sim_k \tilde{u}$, then $\beta_i = \tilde{\beta}_i$

- Follows from case distinctions on $\alpha\beta\alpha$-triples
- Equivalence is completely determined by pairwise equivalence of factors

Theorem (Binary Characterization)

$u, \tilde{u} \in \Sigma_2^*$ with $m := \iota(u) = \iota(\tilde{u}) < k$, then

$$u \sim_k \tilde{u} \iff \forall i \in [m].\beta_i = \tilde{\beta}_i \wedge \forall i \in [m]_0.\alpha_i \sim_{k-m} \tilde{\alpha}_i$$

The Binary Case

Finite Classes

Theorem

Let $w \in \Sigma_2^*$, then $|[w]_{\sim_k}| < \infty$ and in particular $|[w]_{\sim_k}| = 1$, if and only if, $\iota(w) < k$ and $|\alpha_i| < k - \iota(w)$ for all $i \in [\iota(w)]_0$.

[4] Lothaire, *Combinatorics on Words*, Corollary 6.2.8.

The Binary Case

Finite Classes

Theorem

Let $w \in \Sigma_2^*$, then $|[w]_{\sim_k}| < \infty$ and in particular $|[w]_{\sim_k}| = 1$, if and only if, $\iota(w) < k$ and $|\alpha_i| < k - \iota(w)$ for all $i \in [\iota(w)]_0$.

Proof Sketch.

– By Simon and Sakarovitch[4] the classes are singletons or infinite

[4] Lothaire, *Combinatorics on Words*, Corollary 6.2.8.

The Binary Case

Finite Classes

Theorem

Let $w \in \Sigma_2^*$, then $|[w]_{\sim_k}| < \infty$ and in particular $|[w]_{\sim_k}| = 1$, if and only if, $\iota(w) < k$ and $|\alpha_i| < k - \iota(w)$ for all $i \in [\iota(w)]_0$.

Proof Sketch.

- By Simon and Sakarovitch[4] the classes are singletons or infinite
- For words w with $\iota(w) \geq k$ we have $\mathrm{ScatFact}_{\leq k}(w) = \Sigma_2^*$ and thus $w \sim_k w^2$

[4] Lothaire, *Combinatorics on Words*, Corollary 6.2.8.

The Binary Case

Finite Classes

Theorem

Let $w \in \Sigma_2^*$, then $|[w]_{\sim_k}| < \infty$ and in particular $|[w]_{\sim_k}| = 1$, if and only if, $\iota(w) < k$ and $|\alpha_i| < k - \iota(w)$ for all $i \in [\iota(w)]_0$.

Proof Sketch.

- By Simon and Sakarovitch[4] the classes are singletons or infinite
- For words w with $\iota(w) \geq k$ we have $\mathrm{ScatFact}_{\leq k}(w) = \Sigma_2^*$ and thus $w \sim_k w^2$
- If there exists $\alpha_i \in \mathtt{x}^{\geq k-\iota(w)}$, then we can freely add more $\mathtt{x}$ to this α_i since $[\alpha_i]_{\sim_{k-\iota(w)}} = \mathtt{x}^{\geq k-\iota(w)}$ □

[4] Lothaire, *Combinatorics on Words*, Corollary 6.2.8.

The Binary Case

"elementary" MaxSimK

Algorithm 1: MaxSimK for binary words

Input: $u, v \in \Sigma_2^*$

Result: if $u = v$ then ∞ and otherwise the maximum k such that $u \sim_k v$

$(\alpha_0, \beta_1, \ldots, \alpha_{\iota(u)}) := \alpha\beta\text{-Fact}(u)$; // w.r.t. Σ_2

$(\alpha'_0, \beta'_1, \ldots, \alpha'_{\iota(v)}) := \alpha\beta\text{-Fact}(v)$;

if $\iota(u) \neq \iota(v) \vee \text{alph}(u) \neq \text{alph}(v)$ **then** // 2nd condition for $u = \mathtt{x}^i, v = \overline{\mathtt{x}}^j$

  return $\min(\iota(u), \iota(v))$;

else if $\beta_1 = \beta'_1 \wedge \cdots \wedge \beta_{\iota(u)} = \beta'_{\iota(v)}$ **then**

  for $i \in [\iota(u)]_0$ **do** // solve MaxSimK for unary α pairs

    $e_i :=$ **if** $|\alpha_i| = |\alpha'_i|$ **then** ∞ **else** $\min(|\alpha_i|, |\alpha'_i|)$;

  return $\iota(u) + \min\{e_i \mid i \in [\iota(u)]_0\}$;

else

  return $\iota(u)$;

The Binary Case

Enumeration

Idea

- We count classes separated by number of arches
- Extend $\alpha_0\beta_1\cdots\beta_m\alpha_m$ w.r.t $\sim_k$ to $\alpha_0\beta_1\cdots\beta_m\alpha_m\beta_{m+1}\alpha_{m+1}$ w.r.t $\sim_{k+1}$ by choosing suitable β, α
- Choice of β_{m+1} depends only on alph(α_m)

The Binary Case

Enumeration

Idea

- We count classes separated by number of arches
- Extend $\alpha_0\beta_1\cdots\beta_m\alpha_m$ w.r.t $\sim_k$ to $\alpha_0\beta_1\cdots\beta_m\alpha_m\beta_{m+1}\alpha_{m+1}$ w.r.t $\sim_{k+1}$ by choosing suitable β, α
- Choice of β_{m+1} depends only on alph(α_m)

Calculation

- We count spearated by alphabet of last α
- By Cayley-Hamilton we obtain a reccurence satisfied by the matrix

$$|\{w \in \Sigma_2^* \mid \iota(w) = m\}/\sim_k| = \left\| \begin{pmatrix} k-m & k-m & k-m \\ 1 & 2 & 1 \\ k-m & k-m & k-m \end{pmatrix}^m \cdot \begin{pmatrix} k-m \\ 1 \\ k-m \end{pmatrix} \right\|_1 = c_k^m$$

where $c_k^{-1} := 1$, $c_k^0 := 2k+1$ and $c_k^m := 2\cdot(k-m+1)\cdot c_{k-1}^{m-1} - 2\cdot(k-m)\cdot c_{k-2}^{m-2}$.

The Binary Case

Enumeration

Scat Fact Length		Number of Arches								
		0	1	2	3	4	5	6	7	m
	1	3	1							
	2	5	10	1						
	3	7	26	34	1					
	4	9	50	136	116	1				
	5	11	82	358	712	396	1			
	6	13	122	748	2 564	3 728	1 352	1		
	7	15	170	1 354	6 824	18 364	19 520	4 616	1	
	k	$2k+1$								

The Binary Case

Enumeration

Corollary

Let $k \in \mathbb{N}_0$. Over a binary alphabet, the number of congruence classes of $\sim_k$ is given by

$$|\Sigma_2^*/{\sim_k}| = 1 + \sum_{m=0}^{k-1} c_k^m.$$

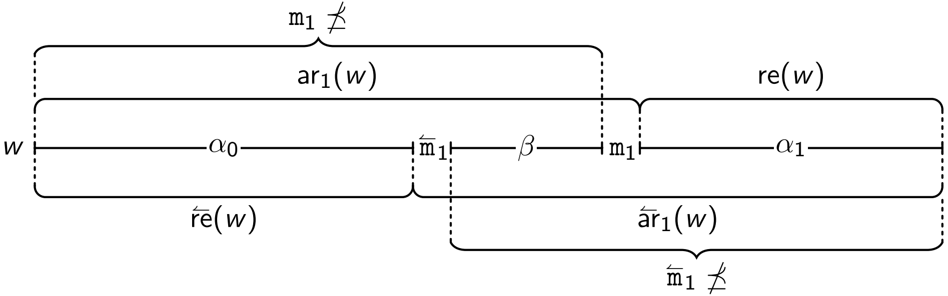
1, 4, 16, 68, 312, 1560, 8528, 50864, 329248, 2298592, 17203264, 137289920, 1162805376, 10409679744, 98146601216, 971532333824, 10068845515264, 108986217860608, 1229343124489216, 14421776248343552, 175640256185075712, 2217046903483447296, 28961312849254387712, 390980190802427064320, ...

Towards the Ternary Case

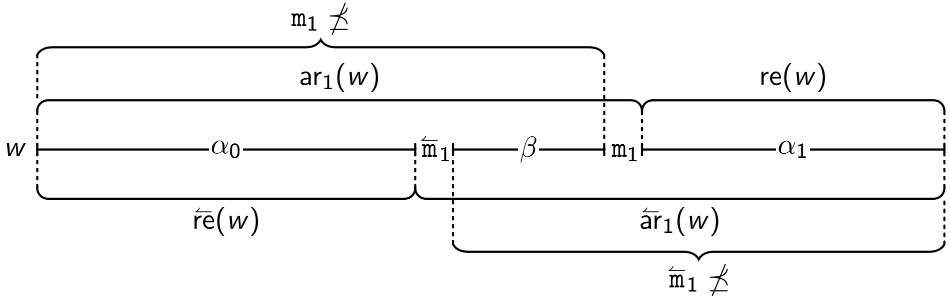
Observations
- In general $w \sim_k \tilde{w}$ does not imply $\beta_i = \tilde{\beta}_i$
- For example $\mathtt{abab} \cdot \mathtt{ab} \cdot \mathtt{c} \sim_3 \mathtt{abab} \cdot \mathtt{ba} \cdot \mathtt{c}$
- Note that $\alpha_0 = \tilde{\alpha}_0$ is "sufficently universal"

Approach
- By the decomposition theorem, it suffices to characterize $\alpha\beta\alpha$-triples
- α-factors are at most binary, handled by induction

Towards the Ternary Case



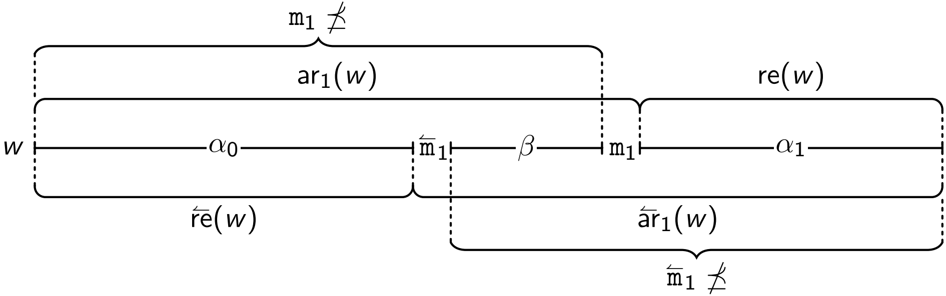
Towards the Ternary Case



Theorem

Let $w \in \Sigma^*$ *with* $\iota(w) = 1$, $k \in \mathbb{N}$, *and* $\overleftarrow{\mathrm{W}} := \{\overleftarrow{\mathtt{m}}_1(\tilde{w}) \mid \tilde{w} \in [w]_{\sim_k}\}$. *If* $|\overleftarrow{\mathrm{W}}| \geq 2$ *then there exists a factorization* $\alpha_0 =: u_1 \cdots u_{k-1}$ *with* $\mathrm{alph}(u_1) \supseteq \ldots \supseteq \mathrm{alph}(u_{k-1}) \supseteq \overleftarrow{\mathrm{W}}$.

Towards the Ternary Case



Theorem

Let $w \in \Sigma^*$ *with* $\iota(w) = 1$, $k \in \mathbb{N}$, *and* $\overleftarrow{\mathrm{W}} := \{\overleftarrow{\mathtt{m}}_1(\tilde{w}) \mid \tilde{w} \in [w]_{\sim_k}\}$. *If* $|\overleftarrow{\mathrm{W}}| \geq 2$ *then there exists a factorization* $\alpha_0 =: u_1 \cdots u_{k-1}$ *with* $\mathsf{alph}(u_1) \supseteq \ldots \supseteq \mathsf{alph}(u_{k-1}) \supseteq \overleftarrow{\mathrm{W}}$.

– Example: $(\mathtt{ababc} \cdot \mathtt{ab} \cdot \mathtt{ab}) \cdot \mathtt{abcd} \cdot \varepsilon \sim_4 (\mathtt{ababc} \cdot \mathtt{ab} \cdot \mathtt{ab}) \cdot \mathtt{bacd} \cdot \varepsilon$

Card	alph(α_0)	alph(α_1)	β RegExp
2–2	$\{a, b\}$	$\{a, c\}$	ba^*c
	$\{a, b\}$	$\{a, b\}$	c
2–1	$\{a, b\}$	$\{c\}$	$(a^+b \mid b^+a)c$
	$\{a, b\}$	$\{a\}$	ba^*c
2–0	$\{a, b\}$	$\emptyset$	$(a^+b \mid b^+a)c$
1–1	$\{a\}$	$\{b\}$	$ab^+c \mid ac^+b \mid ca^+b$
	$\{a\}$	$\{a\}$	ba^*c
1–0	$\{a\}$	$\emptyset$	$ba^*c \mid ab^+c$
0–0	$\emptyset$	$\emptyset$	ab^+c

- $a, b, c \in \Sigma_3$ different; permutation of modi occurs only in case 3 and 5
- exact structure of β determined by $\iota(\alpha_i), \mathrm{alph}(\alpha_i), \mathrm{re}(\alpha_0), \overleftarrow{\mathrm{re}}(\alpha_1)$

References I

- Fleischmann, Pamela et al. "Nearly k-Universal Words – Investigating a Part of Simon's Congruence". In: *Descriptional Complexity of Formal Systems – 24th IFIP WG 1.02 International Conference, DCFS 2022, Proceedings* (Debrecen, Hungary, Aug. 29–31, 2022). Ed. by Yo-Sub Han and György Vaszil. Vol. 13439. Lecture Notes in Computer Science. Springer, 2022, pp. 57–71. URL: https://doi.org/10.1007/978-3-031-13257-5%5C_5.
- Hébrard, Jean-Jacques. "An Algorithm for Distinguishing Efficiently Bit-Strings by their Subsequences". In: *Theoretical Computer Science* 82.1 (1991), pp. 35–49. URL: https://doi.org/10.1016/0304-3975(91)90170-7.
- Karandikar, Prateek, Manfred Kufleitner, and Philippe Schnoebelen. "On the Index of Simon's Congruence for Piecewise Testability". In: *Information Processing Letters* 115.4 (2015), pp. 515–519. URL: https://doi.org/10.1016/j.ipl.2014.11.008.

References II

Lothaire, M. *Combinatorics on Words*. 2nd ed. Cambridge Mathematical Library. Cambridge University Press, 1997. ISBN: 978-0-521-59924-5. URL: https://doi.org/10.1017/CBO9780511566097.