Whitehead's classical algorithm for subgroups

Enric Ventura

Departament de Matemàtica Aplicada III
Universitat Politècnica de Catalunya

&

CRM-Montreal

McGill seminar, Montreal

Oct. 20th, 2010.

Outline

Outline

Definitions and notation

- $A = \{a_1, \ldots, a_k\}$ is a finite alphabet (n letters).
- $A^{\pm 1} = A \cup A^{-1} = \{a_1, a_1^{-1}, \ldots, a_k, a_k^{-1}\}$.
- Usually, $A = \{a, b, c\}$.
- $(A^{\pm 1})^*$ the free monoid on $A^{\pm 1}$ (words on $A^{\pm 1}$).
- $F_A = (A^{\pm 1})^* / \sim$ is the free group on A (words on $A^{\pm 1}$ modulo reduction).
- Every $w \in A^*$ has a unique reduced form,
- 1 denotes the empty word, and $|\cdot|$ the (shortest) length in F_A:
  $|1| = 0, \quad |aba^{-1}| = |abbb^{-1}a^{-1}| = 3, \quad |uv| \leqslant |u| + |v|$.
- $\|\cdot\|$ denotes the (shortest) length in the conjugacy class (i.e. cyclically):
  $\|abbb^{-1}a^{-1}\| = 1$.
- $Aut(F_A)$ and $End(F_A)$ as usual.

Definitions and notation

- $A = \{a_1, \ldots, a_k\}$ is a finite alphabet (n letters).
- $A^{\pm 1} = A \cup A^{-1} = \{a_1, a_1^{-1}, \ldots, a_k, a_k^{-1}\}$.
- Usually, $A = \{a, b, c\}$.
- $(A^{\pm 1})^*$ the free monoid on $A^{\pm 1}$ (words on $A^{\pm 1}$).
- $F_A = (A^{\pm 1})^* / \sim$ is the free group on A (words on $A^{\pm 1}$ modulo reduction).
- Every $w \in A^*$ has a unique reduced form,
- 1 denotes the empty word, and $|\cdot|$ the (shortest) length in F_A:
  $|1| = 0, \quad |aba^{-1}| = |abbb^{-1}a^{-1}| = 3, \quad |uv| \leqslant |u| + |v|$.
- $\|\cdot\|$ denotes the (shortest) length in the conjugacy class (i.e. cyclically):
  $\|abbb^{-1}a^{-1}\| = 1$.
- $Aut(F_A)$ and $End(F_A)$ as usual.

Definitions and notation

- $A = \{a_1, \ldots, a_k\}$ is a finite alphabet (n letters).
- $A^{\pm 1} = A \cup A^{-1} = \{a_1, a_1^{-1}, \ldots, a_k, a_k^{-1}\}$.
- Usually, $A = \{a, b, c\}$.
- $(A^{\pm 1})^*$ the free monoid on $A^{\pm 1}$ (words on $A^{\pm 1}$).
- $F_A = (A^{\pm 1})^* / \sim$ is the free group on A (words on $A^{\pm 1}$ modulo reduction).
- Every $w \in A^*$ has a unique reduced form,
- 1 denotes the empty word, and $|\cdot|$ the (shortest) length in F_A: $|1| = 0, \quad |aba^{-1}| = |abbb^{-1}a^{-1}| = 3, \quad |uv| \leqslant |u| + |v|$.
- $\|\cdot\|$ denotes the (shortest) length in the conjugacy class (i.e. cyclically): $\|abbb^{-1}a^{-1}\| = 1$.
- $Aut(F_A)$ and $End(F_A)$ as usual.

Definitions and notation

- $A = \{a_1, \ldots, a_k\}$ is a finite alphabet (n letters).
- $A^{\pm 1} = A \cup A^{-1} = \{a_1, a_1^{-1}, \ldots, a_k, a_k^{-1}\}$.
- Usually, $A = \{a, b, c\}$.
- $(A^{\pm 1})^*$ the free monoid on $A^{\pm 1}$ (words on $A^{\pm 1}$).
- $F_A = (A^{\pm 1})^* / \sim$ is the free group on A (words on $A^{\pm 1}$ modulo reduction).
- Every $w \in A^*$ has a unique reduced form,
- 1 denotes the empty word, and $|\cdot|$ the (shortest) length in F_A:
  $|1| = 0, \quad |aba^{-1}| = |abbb^{-1}a^{-1}| = 3, \quad |uv| \leqslant |u| + |v|$.
- $\|\cdot\|$ denotes the (shortest) length in the conjugacy class (i.e. cyclically):
  $\|abbb^{-1}a^{-1}\| = 1$.
- $Aut(F_A)$ and $End(F_A)$ as usual.

Definitions and notation

- $A = \{a_1, \ldots, a_k\}$ is a finite alphabet (n letters).
- $A^{\pm 1} = A \cup A^{-1} = \{a_1, a_1^{-1}, \ldots, a_k, a_k^{-1}\}$.
- Usually, $A = \{a, b, c\}$.
- $(A^{\pm 1})^*$ the free monoid on $A^{\pm 1}$ (words on $A^{\pm 1}$).
- $F_A = (A^{\pm 1})^* / \sim$ is the free group on A (words on $A^{\pm 1}$ modulo reduction).
- Every $w \in A^*$ has a unique reduced form,
- 1 denotes the empty word, and $|\cdot|$ the (shortest) length in F_A:
  $|1| = 0, \quad |aba^{-1}| = |abbb^{-1}a^{-1}| = 3, \quad |uv| \leqslant |u| + |v|$.
- $\|\cdot\|$ denotes the (shortest) length in the conjugacy class (i.e. cyclically):
  $\|abbb^{-1}a^{-1}\| = 1$.
- $Aut(F_A)$ and $End(F_A)$ as usual.

Definitions and notation

- $A = \{a_1, \ldots, a_k\}$ is a finite alphabet (n letters).
- $A^{\pm 1} = A \cup A^{-1} = \{a_1, a_1^{-1}, \ldots, a_k, a_k^{-1}\}$.
- Usually, $A = \{a, b, c\}$.
- $(A^{\pm 1})^*$ the free monoid on $A^{\pm 1}$ (words on $A^{\pm 1}$).
- $F_A = (A^{\pm 1})^* / \sim$ is the free group on A (words on $A^{\pm 1}$ modulo reduction).
- Every $w \in A^*$ has a unique reduced form,
- 1 denotes the empty word, and $|\cdot|$ the (shortest) length in F_A: $|1| = 0, \quad |aba^{-1}| = |abbb^{-1}a^{-1}| = 3, \quad |uv| \leqslant |u| + |v|$.
- $\|\cdot\|$ denotes the (shortest) length in the conjugacy class (i.e. cyclically): $\|abbb^{-1}a^{-1}\| = 1$.
- $Aut(F_A)$ and $End(F_A)$ as usual.

Definitions and notation

- $A = \{a_1, \ldots, a_k\}$ is a finite alphabet (n letters).
- $A^{\pm 1} = A \cup A^{-1} = \{a_1, a_1^{-1}, \ldots, a_k, a_k^{-1}\}$.
- Usually, $A = \{a, b, c\}$.
- $(A^{\pm 1})^*$ the free monoid on $A^{\pm 1}$ (words on $A^{\pm 1}$).
- $F_A = (A^{\pm 1})^* / \sim$ is the free group on A (words on $A^{\pm 1}$ modulo reduction).
- Every $w \in A^*$ has a unique reduced form,
- 1 denotes the empty word, and $|\cdot|$ the (shortest) length in F_A:
  $|1| = 0, \quad |aba^{-1}| = |abbb^{-1}a^{-1}| = 3, \quad |uv| \leqslant |u| + |v|$.
- $\|\cdot\|$ denotes the (shortest) length in the conjugacy class (i.e. cyclically):
  $\|abbb^{-1}a^{-1}\| = 1$.
- $Aut(F_A)$ and $End(F_A)$ as usual.

Definitions and notation

- $A = \{a_1, \ldots, a_k\}$ is a finite alphabet (n letters).
- $A^{\pm 1} = A \cup A^{-1} = \{a_1, a_1^{-1}, \ldots, a_k, a_k^{-1}\}$.
- Usually, $A = \{a, b, c\}$.
- $(A^{\pm 1})^*$ the free monoid on $A^{\pm 1}$ (words on $A^{\pm 1}$).
- $F_A = (A^{\pm 1})^* / \sim$ is the free group on A (words on $A^{\pm 1}$ modulo reduction).
- Every $w \in A^*$ has a unique reduced form,
- 1 denotes the empty word, and $|\cdot|$ the (shortest) length in F_A:
  $|1| = 0, \quad |aba^{-1}| = |abbb^{-1}a^{-1}| = 3, \quad |uv| \leqslant |u| + |v|$.
- $\|\cdot\|$ denotes the (shortest) length in the conjugacy class (i.e. cyclically):
  $\|abbb^{-1}a^{-1}\| = 1$.
- $Aut(F_A)$ and $End(F_A)$ as usual.

Definitions and notation

- $A = \{a_1, \ldots, a_k\}$ is a finite alphabet (n letters).
- $A^{\pm 1} = A \cup A^{-1} = \{a_1, a_1^{-1}, \ldots, a_k, a_k^{-1}\}$.
- Usually, $A = \{a, b, c\}$.
- $(A^{\pm 1})^*$ the free monoid on $A^{\pm 1}$ (words on $A^{\pm 1}$).
- $F_A = (A^{\pm 1})^* / \sim$ is the free group on A (words on $A^{\pm 1}$ modulo reduction).
- Every $w \in A^*$ has a unique reduced form,
- 1 denotes the empty word, and $|\cdot|$ the (shortest) length in F_A:
  $|1| = 0, \quad |aba^{-1}| = |abbb^{-1}a^{-1}| = 3, \quad |uv| \leqslant |u| + |v|$.
- $\|\cdot\|$ denotes the (shortest) length in the conjugacy class (i.e. cyclically):
  $\|abbb^{-1}a^{-1}\| = 1$.
- $Aut(F_A)$ and $End(F_A)$ as usual.

Whitehead problem

Whitehead Problem

For a group G, find an algorithm s.t. given $u, v \in G$ decides whether there exists $\varphi \in Aut(G)$ such that $\varphi(u) = v$.

Theorem (Whitehead, 30's)

Whitehead problem is solvable in F_A.

"Proof":
First part: reduce $\|u\|$ and $\|v\|$ as much as possible by applying autos:

$$u \to u_1 \to u_2 \to \cdots \to u',$$

$$v \to v_1 \to v_2 \to \cdots \to v'.$$

Second part: analyze who is image of who by some auto, in the (finite!) sphere of given radius n, $S_n = \{w \in F_k \mid \|w\| = n\}$. □

Whitehead problem

Whitehead Problem

For a group G, find an algorithm s.t. given $u, v \in G$ *decides whether there exists* $\varphi \in Aut(G)$ *such that* $\varphi(u) = v$.

Theorem (Whitehead, 30's)

Whitehead problem is solvable in F_A.

"Proof":

First part: reduce $\|u\|$ and $\|v\|$ as much as possible by applying autos:

$$u \to u_1 \to u_2 \to \cdots \to u',$$

$$v \to v_1 \to v_2 \to \cdots \to v'.$$

Second part: analyze who is image of who by some auto, in the (finite!) sphere of given radius n, $S_n = \{w \in F_k \mid \|w\| = n\}$. □

Whitehead problem

Whitehead Problem

For a group G, find an algorithm s.t. given $u, v \in G$ decides whether there exists $\varphi \in Aut(G)$ such that $\varphi(u) = v$.

Theorem (Whitehead, 30's)

Whitehead problem is solvable in F_A.

"Proof":

First part: reduce $\|u\|$ and $\|v\|$ as much as possible by applying autos:

$$u \to u_1 \to u_2 \to \cdots \to u',$$

$$v \to v_1 \to v_2 \to \cdots \to v'.$$

Second part: analyze who is image of who by some auto, in the (finite!) sphere of given radius n, $S_n = \{w \in F_k \mid \|w\| = n\}$. □

Whitehead problem

Whitehead Problem

For a group G, find an algorithm s.t. given $u, v \in G$ decides whether there exists $\varphi \in Aut(G)$ such that $\varphi(u) = v$.

Theorem (Whitehead, 30's)

Whitehead problem is solvable in F_A.

"Proof":
First part: reduce $\|u\|$ and $\|v\|$ as much as possible by applying autos:

$$u \to u_1 \to u_2 \to \cdots \to u',$$

$$v \to v_1 \to v_2 \to \cdots \to v'.$$

Second part: analyze who is image of who by some auto, in the (finite!) sphere of given radius n, $S_n = \{w \in F_k \mid \|w\| = n\}$. □

Whitehead minimization in polynomial time

Definition

Whitehead automorphisms are those of the form:

$$\begin{array}{rcll} F_A & \to & F_A & \\ a_i & \mapsto & a_i & \textit{(the multiplier)} \\ a_i \neq a_j & \mapsto & a_i^{\epsilon_j} a_j a_i^{\delta_j} & \end{array}$$

where $\epsilon_j = 0, -1$ *and* $\delta_j = 0, 1$ *(there are* $\sim k \cdot 4^k$ *many, where* $k = |A|$*).*

Theorem (Roig, V., Weil, 2007)

There is an algorithm which solves Whitehead Minimization Problem for F_k *in time* $O(n^2 k^3)$*.*

Whitehead minimization in polynomial time

Definition

Whitehead automorphisms are those of the form:

$$\begin{array}{rcll} F_A & \to & F_A & \\ a_i & \mapsto & a_i & \textit{(the multiplier)} \\ a_i \neq a_j & \mapsto & a_i^{\epsilon_j} a_j a_i^{\delta_j} & \end{array}$$

where $\epsilon_j = 0, -1$ *and* $\delta_j = 0, 1$ *(there are* $\sim k \cdot 4^k$ *many, where* $k = |A|$*).*

Theorem (Roig, V., Weil, 2007)

There is an algorithm which solves Whitehead Minimization Problem for F_k *in time* $O(n^2 k^3)$*.*

Gersten's paper

S. Gersten, On Whitehead's algorithm, *Bull. Am. Math. Soc.* **10** (1984) 281-284.

Exactly the same can be done for finitely generated subgroups...
... and the proof will appear somewhere else.

Theorem (Roig, V., Weil, 2007)

There is an algorithm which solves Whitehead Minimization Problem for subgroups $H \leqslant F_k$, in time $O((n^2k^4 + n^3k^2)\log(nk))$, where $n = \|H\|$.

Gersten's paper

S. Gersten, On Whitehead's algorithm, *Bull. Am. Math. Soc.* **10** (1984) 281-284.

Exactly the same can be done for finitely generated subgroups...
... and the proof will appear somewhere else.

Theorem (Roig, V., Weil, 2007)

There is an algorithm which solves Whitehead Minimization Problem for subgroups $H \leqslant F_k$, in time $O((n^2k^4 + n^3k^2)\log(nk))$, where $n = \|H\|$.

Gersten's paper

S. Gersten, On Whitehead's algorithm, *Bull. Am. Math. Soc.* **10** (1984) 281-284.

Exactly the same can be done for finitely generated subgroups...
... and the proof will appear somewhere else.

Theorem (Roig, V., Weil, 2007)

There is an algorithm which solves Whitehead Minimization Problem for subgroups $H \leqslant F_k$, in time $O((n^2k^4 + n^3k^2)\log(nk))$, where $n = \|H\|$.

Gersten's paper

S. Gersten, On Whitehead's algorithm, *Bull. Am. Math. Soc.* **10** (1984) 281-284.

Exactly the same can be done for finitely generated subgroups...
... and the proof will appear somewhere else.

Theorem (Roig, V., Weil, 2007)

There is an algorithm which solves Whitehead Minimization Problem for subgroups $H \leqslant F_k$, *in time* $O((n^2k^4 + n^3k^2)\log(nk))$, *where* $n = \|H\|$.

Outline

Stallings automata

Definition

A Stallings automaton *is a finite A-labeled oriented graph with a distinguished vertex, (X, v), such that:*

1- *X is connected,*

2- *no vertex of degree 1 except possibly v (X is a* core-graph*),*

3- *no two edges with the same label go out of (or in to) the same vertex.*

NO :

YES :

Stallings automata

Definition

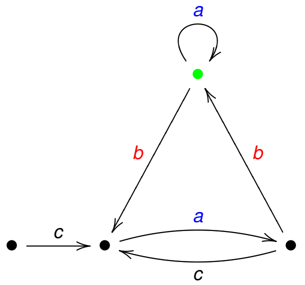
A Stallings automaton is a finite A-labeled oriented graph with a distinguished vertex, (X, v), such that:

1- *X is connected,*

2- *no vertex of degree 1 except possibly v (X is a core-graph),*

3- *no two edges with the same label go out of (or in to) the same vertex.*

NO :



YES :

Stallings automata

Definition

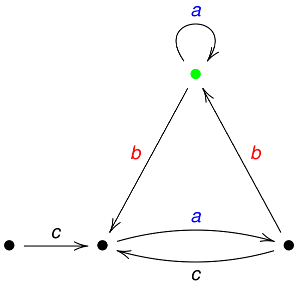
A Stallings automaton *is a finite A-labeled oriented graph with a distinguished vertex,* (X, v)*, such that:*

1- *X is connected,*

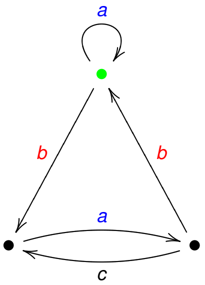
2- *no vertex of degree 1 except possibly v (X is a* core-graph*),*

3- *no two edges with the same label go out of (or in to) the same vertex.*

NO :



YES :



Stallings automata

In the influent paper

J. R. Stallings, Topology of finite graphs, Inventiones Math. 71 (1983), 551-565,

Stallings (building on previous works) gave a bijection between finitely generated subgroups of F_A and Stallings automata:

$$\{\text{f.g. subgroups of } F_A\} \quad \longleftrightarrow \quad \{\text{Stallings automata}\},$$

which is crucial for the modern understanding of the lattice of subgroups of F_A.

Stallings automata

In the influent paper

J. R. Stallings, Topology of finite graphs, Inventiones Math. 71 (1983), 551-565,

Stallings (building on previous works) gave a bijection between finitely generated subgroups of F_A and Stallings automata:

$$\{\text{f.g. subgroups of } F_A\} \quad \longleftrightarrow \quad \{\text{Stallings automata}\},$$

which is crucial for the modern understanding of the lattice of subgroups of F_A.

Stallings automata

In the influent paper

J. R. Stallings, Topology of finite graphs, Inventiones Math. 71 (1983), 551-565,

Stallings (building on previous works) gave a bijection between finitely generated subgroups of F_A and Stallings automata:

$$\{\text{f.g. subgroups of } F_A\} \quad \longleftrightarrow \quad \{\text{Stallings automata}\},$$

which is crucial for the modern understanding of the lattice of subgroups of F_A.

Reading the subgroup from the automata

Definition

To any given (Stallings) automaton (X, v)*, we associate its fundamental group:*

$$\pi(X, v) = \{ \textit{ labels of closed paths at } v \} \leqslant F_A,$$

clearly, a subgroup of F_A*.*

$$\pi(X, \bullet) = \{1, a, a^{-1}, bab, bc^{-1}b, babab^{-1}cb^{-1}, \ldots\}$$

$$\pi(X, \bullet) \not\ni bc^{-1}bcaa$$

Membership problem in $\pi(X, \bullet)$ is solvable.

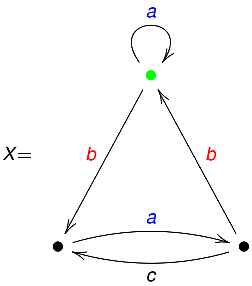
Reading the subgroup from the automata

Definition

To any given (Stallings) automaton (X, v), we associate its fundamental group:

$$\pi(X, v) = \{ \text{labels of closed paths at } v\} \leqslant F_A,$$

clearly, a subgroup of F_A.



$$\pi(X, \bullet) = \{1, a, a^{-1}, bab, bc^{-1}b, babab^{-1}cb^{-1}, \ldots\}$$

$$\pi(X, \bullet) \not\ni bc^{-1}bcaa$$

Membership problem in $\pi(X, \bullet)$ is solvable.

A basis for $\pi(X, v)$

Proposition

For every Stallings automaton (X, v), the group $\pi(X, v)$ is free of rank $rk(\pi(X, v)) = 1 - |VX| + |EX|$.

Proof:

- Take a maximal tree T in X.
- Write $T[p, q]$ for the geodesic (i.e. the unique reduced path) in T from p to q.
- For every $e \in EX - ET$, $x_e = label(T[v, \iota e] \cdot e \cdot T[\tau e, v])$ belongs to $\pi(X, v)$.
- Not difficult to see that $\{x_e \mid e \in EX - ET\}$ is a basis for $\pi(X, v)$.
- And, $$\begin{aligned} |EX - ET| &= |EX| - |ET| \\ &= |EX| - (|VT| - 1) = 1 - |VX| + |EX|. \ \square \end{aligned}$$

A basis for $\pi(X, v)$

Proposition

For every Stallings automaton (X, v), the group $\pi(X, v)$ is free of rank $rk(\pi(X, v)) = 1 - |VX| + |EX|$.

Proof:

- Take a maximal tree T in X.
- Write $T[p, q]$ for the geodesic (i.e. the unique reduced path) in T from p to q.
- For every $e \in EX - ET$, $x_e = label(T[v, \iota e] \cdot e \cdot T[\tau e, v])$ belongs to $\pi(X, v)$.
- Not difficult to see that $\{x_e \mid e \in EX - ET\}$ is a basis for $\pi(X, v)$.
- And, $$\begin{aligned} |EX - ET| &= |EX| - |ET| \\ &= |EX| - (|VT| - 1) = 1 - |VX| + |EX|. \ \square \end{aligned}$$

A basis for $\pi(X, v)$

Proposition

For every Stallings automaton (X, v), the group $\pi(X, v)$ is free of rank $rk(\pi(X, v)) = 1 - |VX| + |EX|$.

Proof:

- Take a maximal tree T in X.
- Write $T[p, q]$ for the geodesic (i.e. the unique reduced path) in T from p to q.
- For every $e \in EX - ET$, $x_e = label(T[v, \iota e] \cdot e \cdot T[\tau e, v])$ belongs to $\pi(X, v)$.
- Not difficult to see that $\{x_e \mid e \in EX - ET\}$ is a basis for $\pi(X, v)$.
- And, $\begin{array}{rcl} |EX - ET| & = & |EX| - |ET| \\ & = & |EX| - (|VT| - 1) = 1 - |VX| + |EX|. \ \square \end{array}$

A basis for $\pi(X, v)$

Proposition

For every Stallings automaton (X, v), the group $\pi(X, v)$ is free of rank $rk(\pi(X, v)) = 1 - |VX| + |EX|$.

Proof:

- Take a maximal tree T in X.
- Write $T[p, q]$ for the geodesic (i.e. the unique reduced path) in T from p to q.
- For every $e \in EX - ET$, $x_e = label(T[v, \iota e] \cdot e \cdot T[\tau e, v])$ belongs to $\pi(X, v)$.
- Not difficult to see that $\{x_e \mid e \in EX - ET\}$ is a basis for $\pi(X, v)$.
- And, $$\begin{aligned} |EX - ET| &= |EX| - |ET| \\ &= |EX| - (|VT| - 1) = 1 - |VX| + |EX|. \ \square \end{aligned}$$

A basis for $\pi(X, v)$

Proposition

For every Stallings automaton (X, v), the group $\pi(X, v)$ is free of rank $rk(\pi(X, v)) = 1 - |VX| + |EX|$.

Proof:

- Take a maximal tree T in X.
- Write $T[p, q]$ for the geodesic (i.e. the unique reduced path) in T from p to q.
- For every $e \in EX - ET$, $x_e = label(T[v, \iota e] \cdot e \cdot T[\tau e, v])$ belongs to $\pi(X, v)$.
- Not difficult to see that $\{x_e \mid e \in EX - ET\}$ is a basis for $\pi(X, v)$.
- And, $|EX - ET| = |EX| - |ET| = |EX| - (|VT| - 1) = 1 - |VX| + |EX|$. □

A basis for $\pi(X, v)$

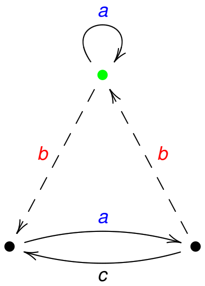
Proposition

For every Stallings automaton (X, v), the group $\pi(X, v)$ is free of rank $rk(\pi(X, v)) = 1 - |VX| + |EX|$.

Proof:

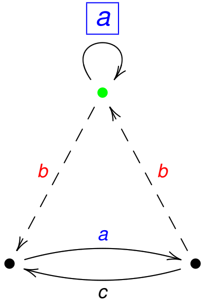
- Take a maximal tree T in X.
- Write $T[p, q]$ for the geodesic (i.e. the unique reduced path) in T from p to q.
- For every $e \in EX - ET$, $x_e = label(T[v, \iota e] \cdot e \cdot T[\tau e, v])$ belongs to $\pi(X, v)$.
- Not difficult to see that $\{x_e \mid e \in EX - ET\}$ is a basis for $\pi(X, v)$.
- And, $$\begin{aligned} |EX - ET| &= |EX| - |ET| \\ &= |EX| - (|VT| - 1) = 1 - |VX| + |EX|. \ \Box \end{aligned}$$

Example



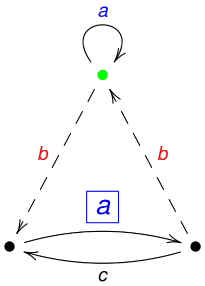
$H = \langle \quad \rangle$

Example



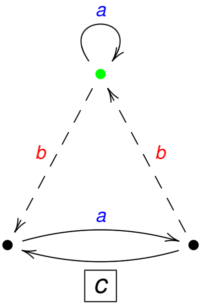
$H = \langle a, \quad \rangle$

Example



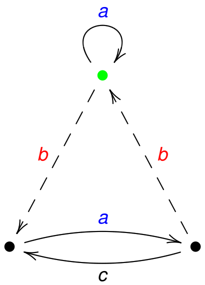
$H = \langle a,\ bab,\ \quad \rangle$

Example



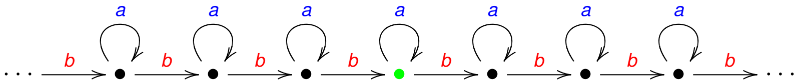
$H = \langle a,\ bab,\ b^{-1}cb^{-1} \rangle$

Example



$H = \langle a,\ bab,\ b^{-1}cb^{-1} \rangle$
$rk(H) = 1 - 3 + 5 = 3.$

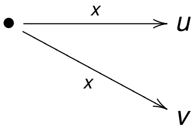
Example-2



$$F_{\aleph_0} \simeq H = \langle \ldots, b^{-2}ab^2, b^{-1}ab, a, bab^{-1}, b^2ab^{-2}, \ldots \rangle \leqslant F_2.$$

Constructing the automata from the subgroup

In any automaton containing the following situation, for $x \in A^{\pm 1}$,



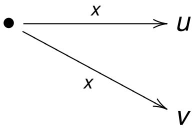
we can fold and identify vertices u and v to obtain

$$\bullet \xrightarrow{\ x\ } u = v \ .$$

This operation, $(X, v) \rightsquigarrow (X', v)$, is called a Stallings folding.

Constructing the automata from the subgroup

In any automaton containing the following situation, for $x \in A^{\pm 1}$,



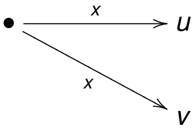
we can fold and identify vertices u and v to obtain

$$\bullet \xrightarrow{\quad x \quad} u = v \ .$$

This operation, $(X, v) \rightsquigarrow (X', v)$, is called a Stallings folding.

Constructing the automata from the subgroup

In any automaton containing the following situation, for $x \in A^{\pm 1}$,



we can fold and identify vertices u and v to obtain

$$\bullet \xrightarrow{\quad x \quad} u = v \ .$$

This operation, $(X, v) \rightsquigarrow (X', v)$, is called a Stallings folding.

Constructing the automata from the subgroup

Lemma (Stallings)

If $(X, v) \rightsquigarrow (X', v')$ is a Stallings folding then $\pi(X, v) = \pi(X', v')$.

Given a f.g. subgroup $H = \langle w_1, \ldots, w_m \rangle \leqslant F_A$ (we assume w_i are reduced words), do the following:

1- *Draw the flower automaton,*

2- *Perform successive foldings until obtaining a Stallings automaton, denoted $\Gamma(H)$.*

Constructing the automata from the subgroup

Lemma (Stallings)

If $(X, v) \rightsquigarrow (X', v')$ is a Stallings folding then $\pi(X, v) = \pi(X', v')$.

Given a f.g. subgroup $H = \langle w_1, \ldots, w_m \rangle \leqslant F_A$ (we assume w_i are reduced words), do the following:

1- *Draw the flower automaton,*

2- *Perform successive foldings until obtaining a Stallings automaton, denoted $\Gamma(H)$.*

Constructing the automata from the subgroup

Lemma (Stallings)

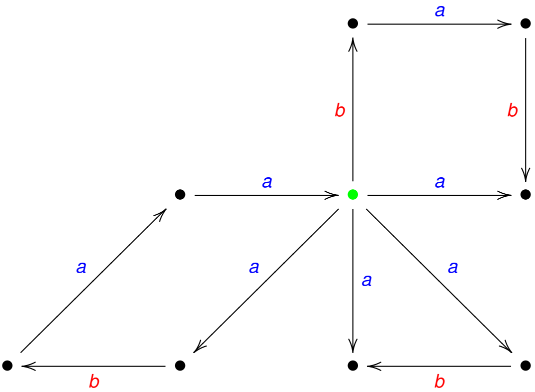
If $(X, v) \rightsquigarrow (X', v')$ is a Stallings folding then $\pi(X, v) = \pi(X', v')$.

Given a f.g. subgroup $H = \langle w_1, \ldots, w_m \rangle \leqslant F_A$ (we assume w_i are reduced words), do the following:

1- *Draw the flower automaton,*

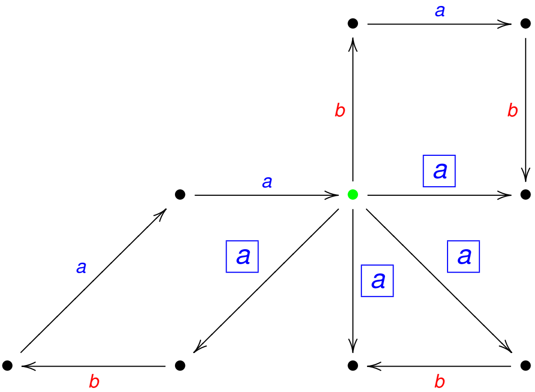
2- *Perform successive foldings until obtaining a Stallings automaton, denoted $\Gamma(H)$.*

Example: $H = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$



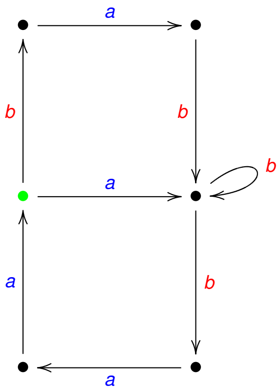
Flower(H)

Example: $H = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$



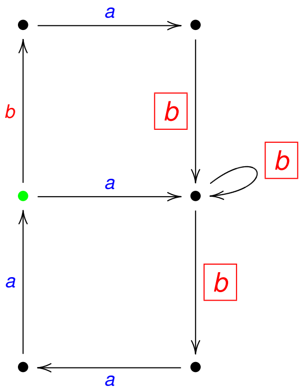
Flower(H)

Example: $H = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$



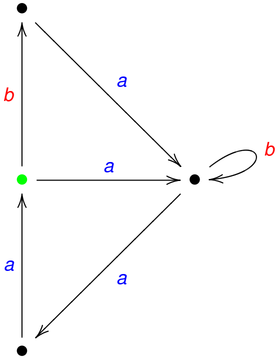
Folding #1

Example: $H = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$



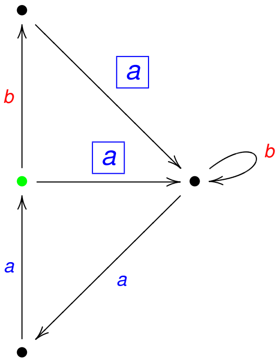
Folding #1.

Example: $H = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$



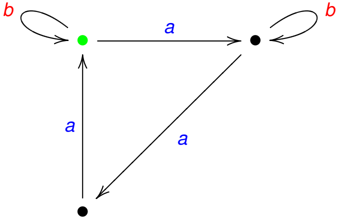
Folding $\#2$.

Example: $H = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$



Folding #2.

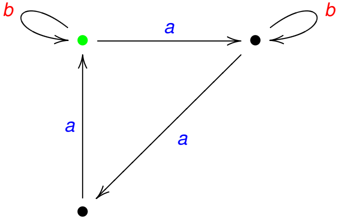
Example: $H = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$



Folding #3. $\Gamma(H)$

By Stallings Lemma, $\pi(\Gamma(H), \bullet) = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$

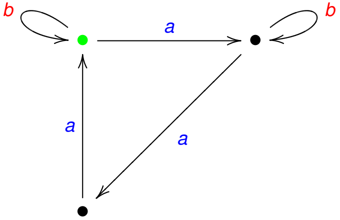
Example: $H = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$



Folding $\#3$. $\Gamma(H)$

By Stallings Lemma, $\pi(\Gamma(H), \bullet) = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$

Example: $H = \langle baba^{-1}, aba^{-1}, aba^2 \rangle$



Folding $\#3$. $\Gamma(H)$

By Stallings Lemma, $\pi(\Gamma(H), \bullet) = \langle baba^{-1}, aba^{-1}, aba^2 \rangle = \langle b, aba^{-1}, a^3 \rangle$

Local confluence

It can be shown that

Proposition

The automaton $\Gamma(H)$ does not depend on the sequence of foldings

Proposition

The automaton $\Gamma(H)$ does not depend on the generators of H.

Theorem

The following is a bijection:

$$\begin{array}{rcl} \{\text{f.g. subgroups of } F_A\} & \longleftrightarrow & \{\text{Stallings automata}\} \\ H & \to & \Gamma(H) \\ \pi(X,v) & \leftarrow & (X,v) \end{array}$$

Local confluence

It can be shown that

Proposition

The automaton $\Gamma(H)$ does not depend on the sequence of foldings

Proposition

The automaton $\Gamma(H)$ does not depend on the generators of H.

Theorem

The following is a bijection:

$$\begin{array}{rcl} \{\text{f.g. subgroups of } F_A\} & \longleftrightarrow & \{\text{Stallings automata}\} \\ H & \to & \Gamma(H) \\ \pi(X,v) & \leftarrow & (X,v) \end{array}$$

Local confluence

It can be shown that

Proposition

The automaton $\Gamma(H)$ does not depend on the sequence of foldings

Proposition

The automaton $\Gamma(H)$ does not depend on the generators of H.

Theorem

The following is a bijection:

$$\begin{array}{rcl} \{\text{f.g. subgroups of } F_A\} & \longleftrightarrow & \{\text{Stallings automata}\} \\ H & \to & \Gamma(H) \\ \pi(X,v) & \leftarrow & (X,v) \end{array}$$

Corollary (Nielsen-Schreier)

Every subgroup of F_A is free.

- Finite automata work for the finitely generated case, but everything extends easily to the general case (using infinite graphs).
- The original proof (1920's) is combinatorial and much more technical.

Nielsen-Schreier Theorem

Corollary (Nielsen-Schreier)

Every subgroup of F_A is free.

- Finite automata work for the finitely generated case, but everything extends easily to the general case (using infinite graphs).
- The original proof (1920's) is combinatorial and much more technical.

Nielsen-Schreier Theorem

Corollary (Nielsen-Schreier)

Every subgroup of F_A is free.

- Finite automata work for the finitely generated case, but everything extends easily to the general case (using infinite graphs).
- The original proof (1920's) is combinatorial and much more technical.

Outline

Peak reduction for subgroups

Definition

For $H \leqslant F_A$*, we define* $\|H\| = \#V(\Gamma'(H))$.

For a cyclic word w*, it is clear that* $\|w\| = \|\langle w\rangle\|$.

Definition

A peak in F_A *is a trio* (H, σ, τ) *where* $H \leqslant_{\mathrm{fg}} F_A$, $\sigma, \tau \in W_I \cup W_{II}$*, such that* $\|\sigma(H)\| \leqslant \|H\|$ *and* $\|\tau(H)\| \leqslant \|H\|$ *with at least one inequality strict.*

Lemma (Peak reduction for subgroups)

For every peak (H, σ, τ) *there exists* $s \geqslant 1$ *and* $\rho_1, \ldots, \rho_s \in W_I \cup W_{II}$ *such that*

- $\tau\sigma^{-1} = \rho_s \cdots \rho_1$,
- $\|\rho_i \cdots \rho_1(H)\| < \|H\|$ *for every* $0 < i < s$.

Peak reduction for subgroups

Definition

For $H \leqslant F_A$, we define $\|H\| = \#V(\Gamma'(H))$.

For a cyclic word w, it is clear that $\|w\| = \|\langle w \rangle\|$.

Definition

A peak in F_A is a trio (H, σ, τ) where $H \leqslant_{\text{fg}} F_A$, $\sigma, \tau \in W_I \cup W_{II}$, such that $\|\sigma(H)\| \leqslant \|H\|$ and $\|\tau(H)\| \leqslant \|H\|$ with at least one inequality strict.

Lemma (Peak reduction for subgroups)

For every peak (H, σ, τ) there exists $s \geqslant 1$ and $\rho_1, \ldots, \rho_s \in W_I \cup W_{II}$ such that

- $\tau\sigma^{-1} = \rho_s \cdots \rho_1$,
- $\|\rho_i \cdots \rho_1(H)\| < \|H\|$ for every $0 < i < s$.

Peak reduction for subgroups

Definition

For $H \leqslant F_A$, we define $\|H\| = \#V(\Gamma'(H))$.

For a cyclic word w, it is clear that $\|w\| = \|\langle w\rangle\|$.

Definition

A peak in F_A is a trio (H, σ, τ) where $H \leqslant_{\rm fg} F_A$, σ, $\tau \in W_I \cup W_{II}$, such that $\|\sigma(H)\| \leqslant \|H\|$ and $\|\tau(H)\| \leqslant \|H\|$ with at least one inequality strict.

Lemma (Peak reduction for subgroups)

For every peak (H, σ, τ) there exists $s \geqslant 1$ and $\rho_1, \ldots, \rho_s \in W_I \cup W_{II}$ such that

- $\tau\sigma^{-1} = \rho_s \cdots \rho_1$,
- $\|\rho_i \cdots \rho_1(H)\| < \|H\|$ *for every* $0 < i < s$.

Peak reduction for subgroups

Definition

For $H \leqslant F_A$, we define $\|H\| = \#V(\Gamma'(H))$.

For a cyclic word w, it is clear that $\|w\| = \|\langle w \rangle\|$.

Definition

A peak in F_A is a trio (H, σ, τ) where $H \leqslant_{\mathrm{fg}} F_A$, $\sigma, \tau \in W_I \cup W_{II}$, such that $\|\sigma(H)\| \leqslant \|H\|$ and $\|\tau(H)\| \leqslant \|H\|$ with at least one inequality strict.

Lemma (Peak reduction for subgroups)

For every peak (H, σ, τ) there exists $s \geqslant 1$ and $\rho_1, \ldots, \rho_s \in W_I \cup W_{II}$ such that

- $\tau\sigma^{-1} = \rho_s \cdots \rho_1$,
- $\|\rho_i \cdots \rho_1(H)\| < \|H\|$ *for every* $0 < i < s$.

Whitehead algorithm for subgroups

Then all the arguments and algorithms extend naturally to this more general context...

Theorem ("Gersten", 1984)

Given two subgroups $H, K \leqslant_{fg} F_A$, it is decidable whether there exists $\varphi \in Aut(F_A)$ such that $\varphi(H) = K$.

Theorem ("Gersten", 1984)

Given a subgroup $H \leqslant_{fg} F_A$, one can algorithmically find $\varphi \in Aut(F_A)$ such that $\|\varphi(H)\|$ is the smallest possible.

Let's do the corresponding improvement with "max-flow min-cut" techniques, to obtain a polynomial time algorithm in this last case ...

Whitehead algorithm for subgroups

Then all the arguments and algorithms extend naturally to this more general context...

Theorem ("Gersten", 1984)

Given two subgroups $H, K \leqslant_{\mathrm{fg}} F_A$, it is decidable whether there exists $\varphi \in Aut(F_A)$ such that $\varphi(H) = K$.

Theorem ("Gersten", 1984)

Given a subgroup $H \leqslant_{\mathrm{fg}} F_A$, one can algorithmically find $\varphi \in Aut(F_A)$ such that $\|\varphi(H)\|$ is the smallest possible.

Let's do the corresponding improvement with "max-flow min-cut" techniques, to obtain a polynomial time algorithm in this last case ...

Whitehead algorithm for subgroups

Then all the arguments and algorithms extend naturally to this more general context...

Theorem ("Gersten", 1984)

Given two subgroups $H, K \leqslant_{\mathrm{fg}} F_A$, it is decidable whether there exists $\varphi \in Aut(F_A)$ such that $\varphi(H) = K$.

Theorem ("Gersten", 1984)

Given a subgroup $H \leqslant_{\mathrm{fg}} F_A$, one can algorithmically find $\varphi \in Aut(F_A)$ such that $\|\varphi(H)\|$ is the smallest possible.

Let's do the corresponding improvement with "max-flow min-cut" techniques, to obtain a polynomial time algorithm in this last case ...

Whitehead algorithm for subgroups

Then all the arguments and algorithms extend naturally to this more general context...

Theorem ("Gersten", 1984)

Given two subgroups $H, K \leqslant_{\rm fg} F_A$, it is decidable whether there exists $\varphi \in Aut(F_A)$ such that $\varphi(H) = K$.

Theorem ("Gersten", 1984)

Given a subgroup $H \leqslant_{\rm fg} F_A$, one can algorithmically find $\varphi \in Aut(F_A)$ such that $\|\varphi(H)\|$ is the smallest possible.

Let's do the corresponding improvement with "max-flow min-cut" techniques, to obtain a polynomial time algorithm in this last case ...

Outline

Whitehead's hypergraph

Thinking a cyclically reduced word w as the circle $\Gamma'(\langle w \rangle)$, its Whitehead graph $Wh(u)$ just describes the in-links of the vertices.

Definition

Let $H \leqslant F_k$ be a f.g. subgroup, and let $\Gamma'(H)$ be its core graph. We define the Whitehead hyper-graph of H, denoted $Wh(H)$, as:

- *vertices: $A^{\pm 1}$,*
- *hyper-edges: for every vertex v in $\Gamma'(H)$, put a hyper-edge consisting on the in-link of v.*

Lemma (Roig, V., Weil, 2007)

Given a f.g. subgroup $H \leqslant F_k$ and a Whitehead automorphism α, think α as a cut in $Wh(H)$, say $\alpha = (T, a)$, and then

$$\|\alpha(H)\| - \|H\| = \mathrm{cap}(T) - \deg(a),$$

where $\mathrm{cap}(T)$ is the number of hyper-edges with at least one vertex in T and one outside T.

Whitehead's hypergraph

Thinking a cyclically reduced word w as the circle $\Gamma'(\langle w \rangle)$, its Whitehead graph $Wh(u)$ just describes the in-links of the vertices.

Definition

Let $H \leqslant F_k$ be a f.g. subgroup, and let $\Gamma'(H)$ be its core graph. We define the Whitehead hyper-graph of H, denoted $Wh(H)$, as:

- *vertices: $A^{\pm 1}$,*
- *hyper-edges: for every vertex v in $\Gamma'(H)$, put a hyper-edge consisting on the in-link of v.*

Lemma (Roig, V., Weil, 2007)

Given a f.g. subgroup $H \leqslant F_k$ and a Whitehead automorphism α, think α as a cut in $Wh(H)$, say $\alpha = (T, a)$, and then

$$\|\alpha(H)\| - \|H\| = \text{cap}(T) - \deg(a),$$

where $\text{cap}(T)$ is the number of hyper-edges with at least one vertex in T and one outside T.

Whitehead's hypergraph

Thinking a cyclically reduced word w as the circle $\Gamma'(\langle w \rangle)$, its Whitehead graph $Wh(u)$ just describes the in-links of the vertices.

Definition

Let $H \leqslant F_k$ be a f.g. subgroup, and let $\Gamma'(H)$ be its core graph. We define the Whitehead hyper-graph of H, denoted $Wh(H)$, as:

- *vertices: $A^{\pm 1}$,*
- *hyper-edges: for every vertex v in $\Gamma'(H)$, put a hyper-edge consisting on the in-link of v.*

Lemma (Roig, V., Weil, 2007)

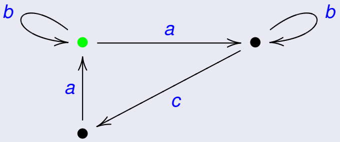
Given a f.g. subgroup $H \leqslant F_k$ and a Whitehead automorphism α, think α as a cut in $Wh(H)$, say $\alpha = (T, a)$, and then

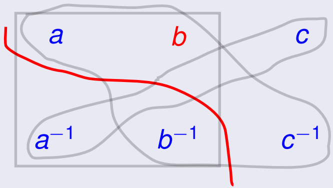
$$\|\alpha(H)\| - \|H\| = \mathrm{cap}(T) - \deg(a),$$

where $\mathrm{cap}(T)$ is the number of hyper-edges with at least one vertex in T and one outside T.

Whitehead's hypergraph

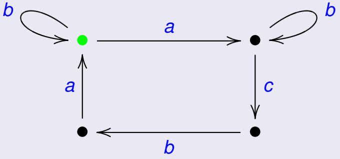
Consider $H = \langle b, aba^{-1}, aca \rangle \leqslant F_3$*. Its core graph* $\Gamma(H)$*, and Whitehead hyper-graph* $Wh(H)$ *are:*





In fact, $\alpha(H) = \langle b, aba^{-1}, acbab \rangle$ *and then*

$\Gamma(\alpha(H)) =$



and so, $4 - 3 = \|\alpha(H)\| - \|H\| = 3 - 2$.

So, Whitehead's Minimization Problem for subgroups reduces to:

- *run over all possible multipliers, say a, (there are $2k$),*
- *find an (a, a^{-1})-cut with minimal possible capacity in the given hyper-graph.*

Unfortunately, there is no analog of max-flow min-cut algorithm for hyper-graphs ...

...but it is still possible to find minimal cuts in polynomial time because of sub-modularity:

Observation

For every f.g. $H \leqslant F_k$, let $W = Wh(H)$ and then the map $\mathcal{P}(A^{\pm 1}) \to \mathbb{N}$, $T \mapsto \mathrm{cap}_W(T)$ is sub-modular.

Minimizing capacities in hyper-graphs

So, Whitehead's Minimization Problem for subgroups reduces to:

- *run over all possible multipliers, say a, (there are $2k$),*
- *find an (a, a^{-1})-cut with minimal possible capacity in the given hyper-graph.*

Unfortunately, there is no analog of max-flow min-cut algorithm for hyper-graphs ...

...but it is still possible to find minimal cuts in polynomial time because of sub-modularity:

Observation

For every f.g. $H \leqslant F_k$, let $W = Wh(H)$ and then the map $\mathcal{P}(A^{\pm 1}) \to \mathbb{N}$, $T \mapsto \mathrm{cap}_W(T)$ is sub-modular.

So, Whitehead's Minimization Problem for subgroups reduces to:

- *run over all possible multipliers, say a, (there are $2k$),*
- *find an (a, a^{-1})-cut with minimal possible capacity in the given hyper-graph.*

Unfortunately, there is no analog of max-flow min-cut algorithm for hyper-graphs ...

...but it is still possible to find minimal cuts in polynomial time because of sub-modularity:

Observation

For every f.g. $H \leqslant F_k$, let $W = Wh(H)$ and then the map $\mathcal{P}(A^{\pm 1}) \to \mathbb{N}$, $T \mapsto \mathrm{cap}_W(T)$ is sub-modular.

Minimizing capacities in hyper-graphs

So, Whitehead's Minimization Problem for subgroups reduces to:

- *run over all possible multipliers, say a, (there are $2k$),*
- *find an (a, a^{-1})-cut with minimal possible capacity in the given hyper-graph.*

Unfortunately, there is no analog of max-flow min-cut algorithm for hyper-graphs ...

...but it is still possible to find minimal cuts in polynomial time because of sub-modularity:

Observation

For every f.g. $H \leqslant F_k$, let $W = Wh(H)$ and then the map $\mathcal{P}(A^{\pm 1}) \to \mathbb{N}$, $T \mapsto \mathrm{cap}_W(T)$ is sub-modular.

Minimizing capacities in hyper-graphs

So, Whitehead's Minimization Problem for subgroups reduces to:

- *run over all possible multipliers, say a, (there are $2k$),*
- *find an (a, a^{-1})-cut with minimal possible capacity in the given hyper-graph.*

Unfortunately, there is no analog of max-flow min-cut algorithm for hyper-graphs ...

...but it is still possible to find minimal cuts in polynomial time because of sub-modularity:

Observation

For every f.g. $H \leqslant F_k$, let $W = Wh(H)$ and then the map $\mathcal{P}(A^{\pm 1}) \to \mathbb{N}$, $T \mapsto \mathrm{cap}_W(T)$ is sub-modular.

Sub-modularity

Definition

A map $f\colon \mathcal{P}(V) \to \mathbb{N}$ *is called* ***sub-modular*** *if, for every* $A, B \subseteq V$, $f(A \cup B) + f(A \cap B) \leqslant f(A) + f(B)$.

Efficient minimization of sub-modular functions is an active research topic in computer science. One of the classical results is the following

Proposition

There exists a algorithm which, given a sub-modular function $f\colon \mathcal{P}(V) \to \mathbb{N}$ *computes its minimum with a number of queries to evaluate* f *bounded above by a polynomial on* $|V|$.

Corollary

There is an algorithm which solves Whitehead Minimization Problem for subgroups $H \leqslant F_k$, *in time* $O((n^2k^4 + n^3k^2)\log(nk))$, *where* $n = \|H\|$.

Sub-modularity

Definition

A map $f\colon \mathcal{P}(V) \to \mathbb{N}$ *is called* *sub-modular* *if, for every* $A, B \subseteq V$, $f(A \cup B) + f(A \cap B) \leqslant f(A) + f(B)$.

Efficient minimization of sub-modular functions is an active research topic in computer science. One of the classical results is the following

Proposition

There exists a algorithm which, given a sub-modular function $f\colon \mathcal{P}(V) \to \mathbb{N}$ *computes its minimum with a number of queries to evaluate* f *bounded above by a polynomial on* $|V|$.

Corollary

There is an algorithm which solves Whitehead Minimization Problem for subgroups $H \leqslant F_k$, *in time* $O((n^2k^4 + n^3k^2)\log(nk))$, *where* $n = \|H\|$.

Sub-modularity

Definition

A map $f\colon \mathcal{P}(V) \to \mathbb{N}$ is called sub-modular if, for every $A, B \subseteq V$,
$f(A \cup B) + f(A \cap B) \leqslant f(A) + f(B)$.

Efficient minimization of sub-modular functions is an active research topic in computer science. One of the classical results is the following

Proposition

There exists a algorithm which, given a sub-modular function $f\colon \mathcal{P}(V) \to \mathbb{N}$ computes its minimum with a number of queries to evaluate f bounded above by a polynomial on $|V|$.

Corollary

There is an algorithm which solves Whitehead Minimization Problem for subgroups $H \leqslant F_k$, in time $O((n^2k^4 + n^3k^2)\log(nk))$, where $n = \|H\|$.

Sub-modularity

Definition

A map $f\colon \mathcal{P}(V) \to \mathbb{N}$ is called sub-modular if, for every $A, B \subseteq V$, $f(A \cup B) + f(A \cap B) \leqslant f(A) + f(B)$.

Efficient minimization of sub-modular functions is an active research topic in computer science. One of the classical results is the following

Proposition

There exists a algorithm which, given a sub-modular function $f\colon \mathcal{P}(V) \to \mathbb{N}$ computes its minimum with a number of queries to evaluate f bounded above by a polynomial on $|V|$.

Corollary

There is an algorithm which solves Whitehead Minimization Problem for subgroups $H \leqslant F_k$, in time $O((n^2k^4 + n^3k^2)\log(nk))$, where $n = \|H\|$.

THANKS