On the Complexity of Conjugacy in Amalgamated Products and HNN Extensions

Von der Fakultät für Informatik, Elektrotechnik und Informationstechnik der Universität Stuttgart zur Erlangung der Würde eines Doktors der Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

Vorgelegt von

Armin Weiß

aus Stuttgart

Hauptberichter: Prof. Dr. Volker Diekert

Mitberichter: Prof. Dr. Markus Lohrey

Tag der mündlichen Prüfung: 3. März 2015

Institut für Formale Methoden der Informatik

2015

Contents

Abstract

This thesis deals with the conjugacy problem in classes of groups which can be written as HNN extension or amalgamated product. The conjugacy problem is one of the fundamental problems in algorithmic group theory which were introduced by Max Dehn in 1911. It poses the question whether two group elements given as words over a fixed set of generators are conjugate. Thus, it is a generalization of the word problem, which asks whether some input word represents the identity. Both, word and conjugacy problem, are undecidable in general.

In this thesis, we consider not only decidability, but also complexity of conjugacy. We consider fundamental groups of finite graphs of groups as defined by Serre – a generalization of both HNN extensions and amalgamated products. Another crucial concept for us are strongly generic algorithms – a formalization of algorithms which work for "most" inputs. The following are our main results:

The elements of an HNN extension which cannot be conjugated into the base group form a strongly generic set if and only if both inclusions of the associated subgroup into the base group are not surjective. For amalgamated products we prove an analogous result.

Following a construction by Stillwell, we derive some undecidability results for the conjugacy problem in HNN extensions with free (abelian) base groups. Next, we show that conjugacy is decidable if all associated subgroups are cyclic or if the base group is abelian and there is only one stable letter. Moreover, in a fundamental group of a graph of groups with free abelian vertex groups, conjugacy is strongly generically in P.

Moreover, we consider the case where all edge groups are finite: If conjugacy can be decided in time $T(N)$ in the vertex groups, then it can be decided in time $\mathcal{O}(\log N \cdot T(N))$ in the fundamental group under some reasonable assumptions on T (here, N is the length of the input). We also derive some basic transfer results for circuit complexity in the same class of groups.

Furthermore, we examine the conjugacy problem of generalized Baumslag-Solitar groups. Our main results are: the conjugacy problem in solvable Baumslag-Solitar groups is TC^0-complete, and in arbitrary generalized Baumslag-Solitar groups it can be decided in $\mathsf{LOGDCFL}$. The uniform conjugacy problem for generalized Baumslag-Solitar groups is hard for $\mathsf{EXPSPACE}$.

Finally, we deal with the conjugacy problem in the Baumslag group, an HNN extension of the Baumslag-Solitar group $\mathbf{BS}_{1,2}$. The Baumslag group has a non-elementary Dehn function, and thus, for a long time, it was considered to have a very hard word problem, until Miaskikov, Ushakov, and Won showed that the word problem, indeed, is in P by introducing a new data structure, the so-called power circuits. We follow their approach and show that the conjugacy problem is strongly generically in P.

Abstract

We conjecture that there is no polynomial time algorithm which works for all inputs, because the divisibility problem in power circuits can be reduced to this conjugacy problem. Also, we prove that the comparison problem in power circuits is complete for P under LOGSPACE reductions.

Zusammenfassung

Diese Arbeit beschäftigt sich mit dem Konjugationsproblem in Gruppen, die sich als HNN-Erweiterung oder amalgamiertes Produkt schreiben lassen. Das Konjugationsproblem ist eines der fundamentalen Probleme der algorithmischen Gruppentheorie, die 1911 von Max Dehn eingeführt wurden. Es ist die Frage, ob zwei Wörter über den Erzeugern der Gruppe in der Gruppe zueinander konjugiert sind. Damit ist es eine Verallgemeinerung des Wortproblems, bei dem für ein Eingabewort festgestellt werden soll, ob es die Identität darstellt. Sowohl das Wort- als auch das Konjugationsproblem sind im Allgemeinen unentscheidbar, wie Boone und Novikov in den 1950er Jahren unabhängig voneinander zeigten.

Bei vielen Beweisen zur Unentscheidbarkeit von algorithmischen Problemen in der Gruppentheorie sind sogenannte HNN-Erweiterungen (benannt nach Higman, Neumann und Neumann) ein wichtiges Hilfsmittel. Daher ist es von besonderem Interesse, unter welchen Bedingungen das Wort- und Konjugationsproblem in HNN-Erweiterungen (effizient) gelöst werden kann. Eine Verallgemeinerung von HNN-Erweiterungen bilden die sogenannten Fundamentalgruppen von Graphen von Gruppen, die Gruppen beschreiben, welche auf Bäumen (nicht-trivial) operieren. Diese Charakterisierung ist die zentrale Aussage der Bass-Serre-Theorie.

In dieser Arbeit werden verschiedene Komplexitätsmaße verwendet, um die Komplexität von Wort- und Konjugationsproblem in Fundamentalgruppen von Graphen von Gruppen zu analysieren: Zeitkomplexität auf RAMs („random access machines“), Schaltkreiskomplexität und Platzkomplexität auf Turingmaschinen einschließlich LOGSPACE-Turingmaschinen, die zusätzlich einen Kellerspeicher verwenden dürfen.

Diese Dissertation ist wie folgt aufgebaut: Nach grundlegenden Definitionen in Kapitel 2 wird in Kapitel 3 eine kurze Einführung in die Bass-Serre-Theorie gegeben. Das Hauptaugenmerk unserer Einführung liegt dabei auf Ersetzungssystemen, insbesondere den sogenannten Britton-Reduktionen. Britton-Reduktionen sind eine effiziente Möglichkeit zur Lösung des Wortproblems von Fundamentalgruppen von Graphen von Gruppen. Auch zur Lösung des Konjugationsproblems sind sie von entscheidender Bedeutung. Außerdem wird hier das Konjugationskriterium für Fundamentalgruppen bewiesen – das wichtigste Werkzeug für alle darauf folgenden Beweise.

Kapitel 4 bildet in gewisser Weise einen Kontrast zum Rest dieser Arbeit. Hier wird der Begriff der Mittelbarkeit („Amenability“) sowie dessen Zusammenhang zu generischen Algorithmen untersucht. Ein generischer Algorithmus berechnet bei den „meisten“ Eingaben die korrekte Lösung (effizient). Es darf aber auch Eingaben geben, bei denen der Algorithmus nicht terminiert.

Zunächst werden einige allgemeine klassische Resultate zur Mittelbarkeit präsentiert – die Beweise dazu sind zum Teil in Anhang B zu finden. Das Hauptresultat in Kapitel 4 ist, dass unter gewissen einfachen Voraussetzungen die Elemente einer Fundamentalgruppe, die nicht in eine der Knotengruppen des Graphen von Gruppen hineinkonjugiert werden können, eine stark generische Menge bilden (stark generisch bedeutet, dass der Anteil der Wörter, die nicht in der Menge sind, exponentiell mit der Länge der Wörter abnimmt). Da in vielen Fällen effiziente Algorithmen zur Lösung des Konjugationsproblems eben für jene Elemente existieren, ist in diesen Fällen das Konjugationsproblem stark generisch effizient lösbar. Es werden zwei Beweise für dieses Resultat präsentiert. Der eine ist eher geometrischer Natur, während der andere mit Random Walks im Schreier-Graphen arbeitet. Letzterer benötigt wenig Hintergrundwissen über Mittelbarkeit, ist dafür aber erheblich länger – deshalb ist er in Anhang B zu finden.

Die nächsten vier Kapitel beschäftigen sich mit Algorithmen für das Konjugationsproblem. In Kapitel 5 werden Fundamentalgruppen von Graphen von Gruppen mit freien bzw. frei abelschen Knotengruppen untersucht. Zunächst werden Unentscheidbarkeitsresultate von Miller verallgemeinert: Sowohl ein amalgamiertes Produkt zweier freier Gruppen vom Rang zwei als auch eine HNN-Erweiterung einer freien Gruppe vom Rang zwei kann ein unentscheidbares Konjugationsproblem haben. Ein analoges Resultat zur Unentscheidbarkeit des Konjugationsproblems gilt für frei abelsche Gruppen. Allerdings sind hier mindestens zwei sogenannte „stable letters“ sowie eine zugrunde liegende Gruppe von größerem Rang notwendig, oder aber mehrere „stable letters“ und eine zugrunde liegende frei abelsche Gruppe vom Rang drei. Andererseits stellt es sich heraus, dass mit nur einem „stable letter“ das Konjugationsproblem in einer HNN-Erweiterung einer frei abelschen Gruppe in jedem Fall entscheidbar ist, was eine gewisse Optimalität beider Resultate zeigt. Ferner ist das Konjugationsproblem in beliebigen Fundamentalgruppen von Graphen von Gruppen mit frei abelschen Knotengruppen generisch in P; es ist immer entscheidbar, wenn alle Kantengruppen zyklisch sind – mit freien oder frei abelschen Knotengruppen.

Wie die Unentscheidbarkeitsresultate zeigen, kann im Allgemeinen von der Entscheidbarkeit des Konjugationsproblems in den Knotengruppen nicht auf die Entscheidbarkeit in der Fundamentalgruppe geschlossen werden. Diese Situation ändert sich, wenn man sich auf endliche Kantengruppen beschränkt, wie man sehr einfach sehen kann (siehe Kapitel 6). Tatsächlich kann in diesem Fall das Konjugationsproblem in der Fundamentalgruppe in Polynomialzeit mit Orakelaufrufen für das Konjugationsproblem der Knotengruppen gelöst werden. Um diese Beobachtung zu verfeinern, wird in Kapitel 6 gezeigt, dass auf einer RAM der Mehraufwand für das Wort- und Konjugationsproblem in der Fundamentalgruppe gegenüber den Knotengruppen maximal logarithmisch ist. Der Beweis für das Wortproblem basiert darauf, dass die notwendigen Britton-Reduktionen in einer speziellen Reihenfolge ausgeführt werden. Das Konjugationsproblem kann durch eine mehrfache Anwendung des String-Matching-Algorithmus von Knuth, Morris und Pratt gelöst werden. Ferner wird in diesem Kapitel mit Hilfe von Schaltkreisen gezeigt, dass das Konjugationsproblem der Fundamentalgruppe in NC^{i+1} ist, falls die Konjugationsprobleme der Knotengruppen in NC^i sind.

In Kapitel 7 wird ein Spezialfall der in Kapitel 5 untersuchten Gruppen behandelt: die sogenannten verallgemeinerten Baumslag-Solitar-Gruppen. Diese sind Fundamentalgruppen endlicher Graphen von Gruppen mit unendlichen zyklischen Knoten- und Kantengruppen. Zuerst werden die auflösbaren Baumslag-Solitar-Gruppen betrachtet, deren Konjugationsproblem TC^0-vollständig ist. Danach geht es um beliebige verallgemeinerte Baumslag-Solitar-Gruppen. Das Hauptresultat des Kapitels ist, dass sowohl das Wort- als auch das Konjugationsproblem verallgemeinerter Baumslag-Solitar-Gruppen in LOGDCFL ist. Hierbei ist LOGDCFL die Klasse jener Sprachen, die sich in LOGSPACE auf eine determinstisch kontextfreie Sprache reduzieren lassen. Betrachtet man den Graph von Gruppen allerdings als Teil der Eingabe, wandelt sich das Bild drastisch: In diesem Fall ist das Konjugationsproblem EXPSPACE-hart.

Schließlich geht es in Kapitel 8 um die Baumslag-(Gersten-)Gruppe, eine HNN-Erweiterung der Baumslag-Solitar-Gruppe $\mathbf{BS}_{1,2}$. Die Baumslag-Gruppe stellte lange Zeit eine besondere algorithmische Herausforderung dar und galt als ein mögliches Bespiel einer Gruppe mit nicht in Elementarzeit lösbarem Wortproblem, bis Miaskikov, Ushakov und Won zeigten, dass ihr Wortproblem tatsächlich in Polynomialzeit lösbar ist. Der Beweis beruht auf einer neuen Datenstruktur, den sogenannten Power Circuits, die in der Lage sind, extrem große Zahlen in komprimierter Form zu speichern.

Mithilfe von Power Circuits kann auch gezeigt werden, dass das Konjugationsproblem der Baumslag-Gruppe generisch in P ist. Allerdings gibt es Eingaben, für die kein Polynomialzeitalgorithmus bekannt ist. Um das Konjugationsproblem für diese Eingaben effizient zu lösen, müsste man das Teilbarkeitsproblem für Power Circuits effizient lösen können. Allerdings kann dies weder durch Probedivision (nach Miaskikov, Ushakov und Won) noch durch Modulo-Rechnen geschehen, da beides zu einem nicht-elementaren Wachstum der Power Circuits führt. Daher ist es ein offenes Problem, ob es einen Polynomialzeitalgorithmus gibt, der das Konjugationsproblem der Baumslag-Gruppe für alle Eingaben löst.

Chapter 1.

Introduction

In 1911, Max Dehn introduced the word problem ("Identitätsproblem"), the conjugacy problem ("Transformationsproblem"), and the isomorphism problem ("Isomorphieproblem") as fundamental problems in algorithmic group theory. The word problem asks whether some word over the generators of the group represents the identity. For the conjugacy problem, the question is whether two given words are conjugate in the group. Here, we focus on the conjugacy problem. Nevertheless, we also have to deal – implicitly or explicitly – with the word problem as it is a special case of the conjugacy problem. This is because some word represents the identity if and only if it is conjugate to the identity.

In recent years, the conjugacy problem has been playing an increasingly more important role in non-commutative cryptography, see e. g. [CJ12, GS09, SZ06]. These applications rely on the fact that it is easy to create elements which are conjugate, but to check whether two given elements are conjugate might be difficult – even if the word problem can be decided efficiently.

In this work, we investigate the conjugacy problem in fundamental groups of graphs of groups. These fundamental groups describe groups acting on trees – a more geometric perspective, shown by Serre in the monograph [Ser77]. For us, this characterization is secondary – we treat fundamental groups of graphs of groups simply as a generalization of HNN extensions and amalgamated products. HNN extensions were first introduced by Higman, Neumann, and Neumann in [HNN49] for the proof of their embedding theorems (which, in particular, state that every countable group can be embedded into a two generator group). Famous examples for HNN extensions are Baumslag-Solitar groups or the Baumslag group, see Chapter 7 and 8. Moreover, HNN extensions became an important tool for undecidability results in group theory: In the 1950s, Novikov [Nov55] and Boone [Boo59] showed independently that there are groups with undecidable word problem (see [BK94] for a survey). Also in Miller's work [Mil71], HNN extensions played a crucial role for the construction of a group with decidable word problem, but undecidable conjugacy problem.

It took some time until also the complexity of algorithmic problems in group theory was studied more carefully. One of the first results showing a low complexity bound is due to Lipton and Zalcstein [LZ77], who proved that the word problem of linear groups (in particular, free groups) is in LOGSPACE. In [Waa81], Waack gave an example of a non-linear group with word problem in LOGSPACE. He also established an analog of the space hierarchy theorem for word problems of groups: if s grows sufficiently faster

than s', then there is a group having word problem in $\mathsf{DSPACE}(s(n)) \setminus \mathsf{DSPACE}(s'(n))$. In [Waa90], the circuit complexity of the word problem is examined. Among other results, it is proven that the word problem in a free product is NC^1-reducible to the word problem in the factors and the word problem of the free group of rank two. In Chapter 6, we generalize this result to other classes of fundamental groups of graphs of groups. The classes $\mathsf{LOGSPACE}$ and NC^1 lie inside the so-called NC hierarchy:

$$\mathsf{AC}^0 \subsetneqq \mathsf{ACC}^0 \subseteq \mathsf{TC}^0 \subseteq \mathsf{NC}^1 \subseteq \mathsf{LOGSPACE} \subseteq \mathsf{AC}^1 \subseteq \mathsf{NC}^2 \subseteq \mathsf{NC}^3 \subseteq \cdots \subseteq \mathsf{NC} \subseteq \mathsf{P}.$$

None except the first of these inclusions is known to be strict. The word problem of finite solvable groups is in ACC^0; for finite non-solvable groups it is complete for NC^1. In his doctoral thesis [Rob93], Robinson obtained the following results: The word problem of solvable Baumslag-Solitar groups and of infinite nilpotent groups is in (non-uniform) TC^0. Moreover, groups of polynomial growth have word problem in NC^1. Finally, Robinson established that the word problem of non-abelian free groups is hard for NC^1. Until now, it is not known whether it is in NC^1, indeed. Here, we extend Robinson's work by examining the word problem and conjugacy problem in all generalized Baumslag-Solitar groups.

Another aspect of complexity is the so-called generic complexity, which was introduced by Kapovich, Myasnikov, Schupp and Shpilrain in [KMSS03, KMSS05]. It formalizes the observation that many problems are easy to solve on most inputs, although for some instances, they are extremely hard to solve. For example a specific form of the halting problem is generically decidable [MR08], but undecidable (in fact, it is strongly generically undecidable). Also in group theory, many problems are generically efficiently decidable. A prominent example for this is the conjugacy problem in Miller's group: although it is undecidable ([Mil71]), it is strongly generically decidable in cubic time [BMR07c]. In this work, we apply generic complexity to the conjugacy problem of the Baumslag group (and some other examples) and show that it is strongly generically in P.

Outline. The outline of this dissertation is as follows: In Chapter 2, we introduce some basic notation and definitions and present the concept of generic algorithms formally. We also define the notion of *doubly* (strongly) generic, which refines "(strongly) generic" for algorithms which receive a pair as input such as the conjugacy problem. Chapter 3 gives a short introduction to Bass-Serre theory. Here, the focus is on rewriting techniques, which are applied in order to derive the Conjugacy Criterion, Theorem 3.19 – our main tool for all following proofs (due to Horadam [Hor81]).

In Chapter 4, we treat the topic of amenability and its relation to strongly generic sets. There are several equivalent definitions for amenability of graphs. We summarize these definitions and recall some classical results – some of the proofs of these results can be found in Appendix B. Then we use these results in order to prove that the elements of an HNN extension which cannot be conjugated into the base group form a strongly generic set if and only if both inclusions of the associated subgroup in the base group are not surjective. For amalgamated products we prove an analogous result.

In Chapter 5, we consider free (abelian) groups as vertex groups of the graph of groups. Following Stillwell's construction of a group with undecidable word problem [Sti82], we construct the following groups all having undecidable conjugacy problem:

- an HNN extension of a free abelian group of rank three with several stable letters (thus, answering a question of Beeker, [Bee11b]).
- an HNN extension of a free abelian group with only two stable letters.
- an HNN extension of a free group of rank two with only one stable letter.
- an amalgamated product of two free groups of rank two.

Next, we show that conjugacy is decidable if all associated subgroups are cyclic or if the base group is abelian and there is only one stable letter. Moreover, in a fundamental group of a graph of groups with free abelian vertex groups, conjugacy is doubly strongly generically in P.

Chapter 6 considers the case where all edge groups are finite. It is straightforward to see that if the conjugacy problem in the vertex groups is decidable in some complexity class $\mathcal{C}$, then the conjugacy problem in the fundamental group is decidable in $\mathsf{P}^{\mathcal{C}}$ (polynomial time with oracles for $\mathcal{C}$). When considering the complexity on a random access machine (RAM), we obtain a stronger result: If conjugacy can be decided in time $T(N)$ in the vertex groups, then it can be decided in time $\mathcal{O}(\log N \cdot T(N))$ in the fundamental group under some reasonable assumptions on T (here, N is the length of the input). Furthermore, we examine the circuit complexity of conjugacy in fundamental groups of finite graphs of groups with finite edge groups and generalize Waack's result for free products in two ways: First, we show that the conjugacy problem in the fundamental group can be solved by a NC^2 circuit with oracle gates for the conjugacy problem of the vertex groups. Then, we show that in a even more restricted class of graphs of groups, indeed, the conjugacy problem in the fundamental group is NC^1-Turing-reducible to conjugacy in the vertex groups.

In Chapter 7, we examine the conjugacy problem of generalized Baumslag-Solitar groups more thoroughly. Our main results are that conjugacy in solvable Baumslag-Solitar groups is TC^0-complete, and in all generalized Baumslag-Solitar groups, it can be decided in $\mathsf{LOGDCFL}$ (the class of languages which are $\mathsf{LOGSPACE}$-reducible to a deterministic context-free language – a subclass of AC^1). If the graph of groups which represents the generalized Baumslag-Solitar group is part of the input, the conjugacy problem is $\mathsf{EXPSPACE}$-hard because the uniform word problem of commutative semigroups can be reduced to it.

Finally, we introduce the Baumslag group, an HNN extension of the Baumslag-Solitar group $\mathbf{BS}_{1,2}$. Chapter 8 deals with the conjugacy problem in this group. The Baumslag group has a non-elementary Dehn function, and thus, for a long time, it was considered to have a very hard word problem. Nevertheless, Miaskikov, Ushakov, and Won could show that the word problem is in P by introducing a new data structure, the so-called power circuits. We follow this approach and show that also the conjugacy problem is doubly strongly generically in P. We are unable to give a polynomial time algorithm for the worst case and we conjecture that there is no such algorithm. This is

because an elementary-time solution for conjugacy in the Baumslag group would yield an elementary-time algorithm for the divisibility problem in power circuits. However, we show that standard methods for the solution of the divisibility problem cannot work in elementary time. Furthermore, we prove that the comparison problem in power circuits is complete for P under LOGSPACE reductions.

Chapter 2.

Preliminaries

This chapter is dedicated to introducing the necessary background and fix notation. Parts of it are taken from [DW13]. The reader who is familiar with the concepts introduced here, might skip this chapter and consult the definitions whenever needed. In order to have a concise presentation, we omit some of the basic definitions of complexity theory. These can be found in Appendix A.

Throughout, we assume $0 \in \mathbb{N}$. The logarithm log is always to base two if not specified otherwise. For some number $x \in \mathbb{R}$, its integral part (i. e., the largest integer less or equal than x) is denoted by $\lfloor x \rfloor$. We denote the cardinality of a set A by $|A|$. For some function φ we write $\mathrm{dom}(\varphi)$ for the domain of φ. The free group of rank m is denoted by F_m. If H is a subgroup of a group G, we write $H \leq G$.

2.1. Rewriting Systems

Let X be a set; a *rewriting system* over X is a binary relation $\Longrightarrow \subseteq X \times X$. If $(x, y) \in \Longrightarrow$, we write $x \Longrightarrow y$. The idea of the notation is that $x \Longrightarrow y$ indicates that we can rewrite x in one step into the element y. We denote the reflexive and transitive closure of $\Longrightarrow$ by $\overset{*}{\Longrightarrow}$; and by $\overset{*}{\Longleftrightarrow}$ its reflexive, transitive, and symmetric closure. The relation $\overset{*}{\Longleftrightarrow}$ is the smallest equivalence relation such that x and y are in the same class for all $x \Longrightarrow y$. We also write $y \Longleftarrow x$ if $x \Longrightarrow y$. The rewriting system $\Longrightarrow$ is called

- *confluent*, if $y \overset{*}{\Longleftarrow} x \overset{*}{\Longrightarrow} z$ implies $\exists\, w : y \overset{*}{\Longrightarrow} w \overset{*}{\Longleftarrow} z$,
- *Church-Rosser*, if $y \overset{*}{\Longleftrightarrow} z$ implies $\exists\, w : y \overset{*}{\Longrightarrow} w \overset{*}{\Longleftarrow} z$,
- *locally confluent*, if $y \Longleftarrow x \Longrightarrow z$ implies $\exists\, w : y \overset{*}{\Longrightarrow} w \overset{*}{\Longleftarrow} z$,
- *terminating* or *Noetherian*, if there are no infinite chains

$$x_0 \Longrightarrow x_1 \Longrightarrow \cdots x_{i-1} \Longrightarrow x_i \Longrightarrow \cdots,$$

- *convergent*, if it is locally confluent and terminating.

Two main properties of rewriting systems are stated in the following classical theorem.

Theorem 2.1 ([BO93, Jan88])**.**

(i) *Confluence is equivalent to the Church-Rosser property.*

(ii) *A locally confluent and terminating system is confluent. Thus, a convergent system satisfies the Church-Rosser property.*

2.2. Presentations of Monoids and Groups

An *alphabet* is simply a set; its elements are called *letters*. A *word* w is an element in some k-fold Cartesian product Σ^k, where k is the *length* $|w|$ of w (not to be confused with the cardinality of a set A which is also denoted as $|A|$). Frequently, we write $w = a_1 \cdots a_k$ to denote a word of length k. For $a \in \Sigma$, we write $|w|_a = |\{i \in \{1, \ldots, k\} \mid a_i = a\}|$ for the number of occurrences of the letter a in the word w. The union $\Sigma^* = \bigcup_{k \geq 0} \Sigma^k$ is a monoid by

$$(a_1 \cdots a_k) \cdot (b_1 \cdots b_\ell) = a_1 \cdots a_k b_1 \cdots b_\ell.$$

The neutral element is the *empty word*, which is denoted as 1. It is the unique word of length 0. The monoid Σ^* is *free* over Σ because mappings from Σ to a monoid M are in canonical one-to-one correspondence with homomorphisms from Σ^* to M. If $\eta : \Sigma^* \to M$ is surjective, then we call η a *presentation* of M (by Σ^*).

Let T be a semigroup and $S \subseteq T \times T$ be a set of pairs. This defines a rewriting system $\underset{S}{\Longrightarrow}$ over T by $x \underset{S}{\Longrightarrow} y$ if $x = u\ell v$ and $y = urv$ for some $(\ell, r) \in S$. It is common to denote a rule $(\ell, r) \in S$ by $\ell \to r$. Since $\overset{*}{\underset{S}{\Longleftrightarrow}}$ is an equivalence relation, we can form the set of equivalence classes $T/S = \{[x] \mid x \in T\}$, where $[x] = \left\{ y \in T \;\middle|\; x \overset{*}{\underset{S}{\Longleftrightarrow}} y \right\}$. Now, T/S becomes a semigroup by $[x] \cdot [y] = [xy]$, and the mapping $x \to [x]$ yields a canonical homomorphism $\eta : T \to T/S$.

By a slight abuse of language, we call $S \subseteq T \times T$ itself a rewriting system and transfer the properties of Section 2.1, like confluence, from $\underset{S}{\Longrightarrow}$ to S. An element $x \in T$ of the semigroup T is called *irreducible* (w. r. t. S) or *S-reduced* if x cannot be written in the form $x = u\ell v$ for any $\ell \to r \in S$ and $u, v \in T$. The set of all irreducible elements is denoted by $\mathrm{IRR}(S)$. If the rewriting system S is convergent, then for every $x \in T$ there is exactly one element $\hat{x} \in T$ with $x \overset{*}{\underset{S}{\Longrightarrow}} \hat{x} \in \mathrm{IRR}(S)$. In this case, the canonical homomorphism $\eta : T \to T/S$ induces a bijection between $\mathrm{IRR}(S)$ and T/S; hence, $\mathrm{IRR}(S)$ becomes a set of *normal forms* for the quotient semigroup T/S.

In case we have $S \subseteq \Sigma^* \times \Sigma^*$, we call S a *semi-Thue system.* Thus, a semi-Thue system defines a quotient monoid $M = \Sigma^*/S$ and a natural presentation $\eta : \Sigma^* \to \Sigma^*/S$. If we can choose Σ to be finite, then M is called *finitely generated* (f. g.), and if, in addition, we can choose $S \subseteq \Sigma^* \times \Sigma^*$ to be finite, then M is called *finitely presented*.

Throughout, the convention is that for two words $v, w \in \Sigma^*$ we write $v =_M w$ if $\eta(v) = \eta(w)$; likewise for some subset $A \subseteq M$ we write $w \in_M A$ if $\eta(w) \in A$. Sometimes

we also write "$v = w$ in M" instead of $v =_M w$. Otherwise, $=$ and $\in$ denote equality and inclusion of words. If $a, b \in \Sigma$ and $c \in \Sigma \cup \{1\}$ with $ab =_M c$, then we write $[ab] = c$ (equality as words). Most of the times we do not write the homomorphism η and we consider Σ as a subset of Σ^*/S. We call Σ a *generating set* of Σ^*/S.

Example 2.2. Let Σ be a set and $\overline{\Sigma} = \{\overline{a} \mid a \in \Sigma\}$ be a disjoint copy of Σ. We extend $a \mapsto \overline{a}$ to an involution without fixed points on $\widetilde{\Sigma} = \Sigma \cup \overline{\Sigma}$ by $\overline{\overline{a}} = a$. The system $S = \left\{ a\overline{a} \to 1 \;\middle|\; a \in \widetilde{\Sigma} \right\}$ is convergent; it defines the *free group* $F_\Sigma = \widetilde{\Sigma}^*/S$ with basis Σ (i. e., of rank $|\Sigma|$). The set $\mathrm{IRR}(S)$ is the set of *freely reduced* normal forms.

If $\Sigma^*/S = G$ is a group, we sometimes call Σ a *monoid generating set* in order to emphasize that Σ generates Σ^*/S as a monoid and not only as a group. In the special case that $\Sigma = \Sigma^{-1}$ as a subset of G, we call Σ a *symmetric generating set*. If Σ is a symmetric generating set and $a \in \Sigma$, then we denote the inverse letter with $\overline{a} \in \Sigma$ as in Example 2.2. Moreover, we extend this involution to words by setting $\overline{v \cdot w} = \overline{w} \cdot \overline{v}$ for $v, w \in \Sigma^*$. That means we have $\overline{w} =_G w^{-1}$ for all $w \in \Sigma^*$. For simplicity, we often use $\overline{a}$ and a^{-1} interchangeably, e. g., we write $a^{-\ell}$ instead of $\overline{a}^\ell$ for $\ell \in \mathbb{N}$.

If G is generated as group by a set $\Sigma \subseteq G$, we obtain a symmetric generating set $\widetilde{\Sigma}$ as in the case of the free group (Example 2.2) by setting $\widetilde{\Sigma} = \Sigma \cup \overline{\Sigma}$ (with the involution as above). We have $G = F_\Sigma/N$ where N is some normal subgroup. If $R \subseteq F_\Sigma$ is some subset such that the normal closure of R is N, we write

$$G = \langle\, \Sigma \mid R \,\rangle\,.$$

The set R defines a rewriting system $S \subseteq \widetilde{\Sigma} \times \widetilde{\Sigma}$ as follows: For every $r \in R$ we choose some arbitrary $w_r \in \widetilde{\Sigma}^*$ with $w_r =_{F_\Sigma} r$ and we set

$$S = \left\{ a\overline{a} \to 1, w_r \to 1 \;\middle|\; a \in \widetilde{\Sigma}, r \in R \right\}.$$

We have $G = F_\Sigma/N = \widetilde{\Sigma}^*/S$, and G is called *finitely generated* resp. *finitely presented* if Σ^*/S is so. If $G = \langle\, \Sigma \mid R \,\rangle$ is a group, Δ some alphabet, and $R' \subseteq G * F_\Delta$ a set of relations ($G * F_\Delta$ is the free product), we define $\langle\, G, \Delta \mid R' \,\rangle = \langle\, \Sigma \cup \Delta \mid R \cup R' \,\rangle$.

2.3. Algorithmic Problems in Group Theory

Algorithmic problems in group theory were first introduced by Dehn in [Deh11]. We are interested in the following problems for a group G with monoid generating set Σ:

- *word problem*: $\mathrm{WP}(G) = \{w \in \Sigma^* \mid w =_G 1\}$,
- *subgroup membership problem* of H in G: $\{w \in \Sigma^* \mid w \in_G H\}$,
- *conjugacy problem*: $\mathrm{CP}(G) = \{(v, w) \in \Sigma^* \times \Sigma^* \mid \exists\, z \in G : zvz^{-1} =_G w\}$.

The word problem is a special case of the conjugacy problem since v is conjugate to 1 (denoted by $v \sim_G 1$) if and only if $v =_G 1$. Also the subgroup membership problem is a generalization of the word problem; therefore, it is sometimes called *generalized word problem* in literature.

Note that the complexity of these problems is invariant under change of generating sets (given that the complexity class is closed under inverse homomorphisms – a fairly natural assumption). Hence, we can talk about the complexity of these problems as a property of the group.

2.4. Graphs

This section fixes notation for graphs according to the book [Ser80]. A *directed graph* $\Gamma = (V, E, \iota, \tau)$ is given by the following data: A set of *vertices* $V = V(\Gamma)$ and a set of *edges* $E = E(\Gamma)$ together with two mappings $\iota, \tau : E \to V$. The vertex $\iota(e)$ is the *source* of e, and $\tau(e)$ is the *target* of e. A vertex u and an edge e are *incident* if $u \in \{\iota(e), \tau(e)\}$. If $\tau(e) = u$ (resp. $\iota(e) = u$), we call e an *incoming edge* (resp. *outgoing edge*) of u. The *in-degree* (resp. *out-degree*) of a vertex $u \in V(\Gamma)$ is the number of incoming edges (resp. outgoing edges); Γ is called *locally finite* if the in-degrees and out-degrees of all vertices are finite. Furthermore, Γ is said to have *bounded degree* if there is some constant K such that the in- and out-degrees of all vertices are less than K. If both the in-degrees and out-degrees of all vertices are equal to some constant d, then Γ is called *d-regular*. If there is some d such that Γ is d-regular, then Γ is called *regular*. A directed graph is finite, if it has finitely many vertices and edges. A directed graph $\Gamma' = (V', E', \iota', \tau')$ is a *subgraph* of $\Gamma = (V, E, \iota, \tau)$, if $V' \subseteq V$, $E' \subseteq E$ and ι' and τ' are the restrictions of ι and τ to E'.

An *undirected graph* $\Gamma = (V, E, \iota, \tau, \bar{\cdot})$ is a directed graph such that the set of edges E is equipped with an involution $e \mapsto \bar{e}$ without fixed points such that $\iota(e) = \tau(\bar{e})$. An *undirected edge* is the set $\{e, \bar{e}\}$. For an undirected graph Γ, the *degree* of some vertex is defined as its in-degree (= out-degree). In the following, a *graph* always means an undirected graph; otherwise, we say specifically "directed graph". Also, we assume all graphs to be locally finite. For simplicity, we often suppress the incidence functions (and involution) and write only $\Gamma = (V, E)$ for a (directed) graph Γ knowing that the incidence functions (and involution) also belong to Γ.

Graphs without *loops* (edges e with $\iota(e) = \tau(e)$) and *multi-edges* (edges $e \neq f$ with $\iota(e) = \iota(f)$ and $\tau(e) = \tau(f)$) are called *simple*. For simple graphs, we identify the set of edges with some subset of $\binom{V(\Gamma)}{2}$, i. e., we write $\{u, v\} \in E$ for an edge $e \in E$ with $\{\iota(e), \tau(e)\} = \{u, v\}$. Likewise, we define simple directed graphs; in this case we consider $E \subseteq V \times V$.

A *path* of length n in a (directed) graph is a subgraph $(\{v_0, \ldots, v_n\}, \{e_1, \ldots, e_n\})$ such that $\iota(e_i) = v_{i-1}$ and $\tau(e_i) = v_i$ for all $1 \leq i \leq n$. Sometimes we suppress the vertices and call the sequence of edges a path. A path is called *without backtracking* if $e_i \neq \bar{e}_{i+1}$ for all i. A *cycle* is a path without backtracking with $n \geq 1$, $v_0 = v_n$, and such that the vertices $v_1, \ldots, v_n$ are pairwise distinct. The *distance* $d(u, v)$ between vertices u

and v is defined as the length of a shortest path connecting u and v. We let $d(u,v) = \infty$ if there is no such path. An undirected graph Γ is called *connected* if $d(u,v) < \infty$ for all vertices u and v. With this definition, every undirected graph becomes a metric space.

A *tree* is a connected simple graph without any cycle. In particular, a tree is undirected. A tree T is called *spanning tree* of a graph Γ if $V(T) = V(\Gamma)$ and $E(T) \subseteq E(\Gamma)$, i. e., the tree T connects all vertices of Γ. A directed graph without any cycle is called a *directed acyclic graph* (*DAG*).

Morphisms and Group Actions. An action of a group G on a graph Γ consists of actions of G on the vertex set and on the edge set which respect the incidence functions and involution, i. e., for all $g \in G$, $e \in E(\Gamma)$ we have $\overline{g \cdot e} = g \cdot \bar{e}$ and $\iota(g \cdot e) = g \cdot \iota(e)$. We say the action is *without edge inversion* if for all $e \in E(\Gamma)$ and $g \in G$ we have $ge \neq \bar{e}$. If G acts on Γ without edge inversion, then we can define the quotient graph ${}_G\backslash\Gamma$: Its vertices (resp. edges) are the *orbits* $G \cdot u$ for $u \in V(\Gamma)$ (resp. $G \cdot e$ for $e \in E(\Gamma)$) with incidences $\iota(G \cdot e) = G \cdot \iota(e)$ and $\tau(G \cdot e) = G \cdot \tau(e)$.

An action of some group G on a set X is called *faithful* if there is no group element except 1 leaving all elements of X invariant; it is called *free* if the existence of some $x \in X$ with $gx = x$ implies $g = 1$, i. e., all stabilizers $G_x = \{g \in G \mid gx = x\}$ for $x \in X$ are trivial. An action of a group on a graph is called free if it is without edge inversion and the action on the set of vertices is free.

Cayley Graphs. Let G be a group with 1 as neutral element. Let Σ be a symmetric generating set of G. The *Cayley graph* $\Gamma = \Gamma(G, \Sigma)$ of G (with respect to Σ) is defined by $V(\Gamma) = G$ and $E(\Gamma) = G \times \Sigma$ with the incidence functions $s(g,a) = g$, $t(g,a) = ga$ and involution $\overline{(g,a)} = (ga, \bar{a})$. As Σ generates G, the Cayley graph Γ is connected. It is locally finite if and only if Σ is finite. In the following, we always assume that Σ is finite. There is a natural action of G on $\Gamma(G, \Sigma)$.

2.5. Complexity

We apply various measures for complexity: time complexity on random access machines, space complexity on Turing machines – including DLPAuxPDAs, some extension of Turing machines – and circuit complexity. We use standard $\mathcal{O}$ notation for functions from $\mathbb{N}$ or $\mathbb{R}_{\geq 0}$ to the non-negative reals $\mathbb{R}_{\geq 0}$ (i. e., $\mathcal{O}(f)$ denote the class of functions growing asymptotically at most as fast as f).

An algorithm $\mathcal{A}$ computes a function between domains D and D'. For decision problems (i. e., formal languages), we have $D' = \{0,1\}$. In our applications, D is a union $D = \bigcup \left\{ D^{(N)} \;\middle|\; N \in \mathbb{N} \right\}$ where each $D^{(N)}$ is finite – it consists of the elements of "length" at most N. Usually, we use $D = \Sigma^*$ for some finite alphabet Σ with $D^{(N)} = \Sigma^{\leq N}$, or $D = \Sigma^* \times \Sigma^*$ with $D^{(N)} = \{(v,w) \mid |v| + |w| \leq N\}$. Sometimes also the set of freely reduced words over some symmetric alphabet is used instead of Σ^* – with $D^{(N)}$ being the set of freely reduced words of length N.

For time complexity, we use random access machines (RAMs) as in [Pap94, Sec. 2.6]. A short description of our RAM model can be found in Appendix A. Let $\mathcal{A}$ be some algorithm; the time complexity $T_{\mathcal{A}}$ is defined by $T_{\mathcal{A}}(N) = \max\left\{T_{\mathcal{A}}(w) \;\middle|\; w \in D^{(N)}\right\}$ where $T_{\mathcal{A}}(w)$ denotes the number of steps performed by the RAM executing $\mathcal{A}$ on input w. A crucial fact is that, up to polynomial transformations, time complexity on RAMs is equivalent to time complexity on Turing machines.

Average-Case Complexity. Assuming a uniform distribution among elements in $D^{(N)}$, the *average-case complexity* of some algorithm $\mathcal{A}$ is defined by

$$\mathrm{av}_{\mathcal{A}}(N) = \frac{1}{|D^{(N)}|} \sum_{w \in D^{(N)}} T_{\mathcal{A}}(w).$$

Non-elementary and Elementary Time. The *tower function* $\mathrm{tow} : \mathbb{N} \to \mathbb{N}$ is defined by $\mathrm{tow}(0) = 0$ and $\mathrm{tow}(i+1) = 2^{\mathrm{tow}(i)}$ for $i \geq 0$. It is primitive recursive, but $\mathrm{tow}(7)$ written in binary cannot be stored in the memory of any conceivable real-world computer.

We say that a function $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is *elementary*, if the growth of f can be bounded by a fixed number of exponentials. It is called *non-elementary* if it is not elementary, but $f(n) \in \mathrm{tow}(\mathcal{O}(n))$. Thus, non-elementary means a lower and an upper bound. We say an algorithm is *elementary* or runs in *elementary time* if its running time (or equivalently space) is bounded by some elementary function. An algorithm is called *non-elementary* if its running time is bounded by some non-elementary function but not by an elementary function.

Complexity on Turing Machines. We use the standard classes for time and space complexity on Turing machines. In particular, the class $\mathsf{LOGSPACE}$ is the class of formal languages which are accepted by a deterministic Turing machine (with read-only input tape) using $\mathcal{O}(\log N)$ space on the work tape where N is the length of the input. Other complexity classes we consider are P for polynomial time, and $\mathsf{EXPSPACE} = \bigcup_{k \in \mathbb{N}} \mathsf{DSPACE}(2^{n^k})$ for exponential space.

We also use oracle Turing machines (for details see Appendix A). Let $\mathcal{C}$ be some complexity class defined by Turing machines and let $\mathcal{D}$ be some arbitrary class of languages. Then a language L is in $\mathcal{C}^{\mathcal{D}}$ if and only if there is a Turing machine which uses oracles for a finite subset of languages in $\mathcal{D}$ and which accepts L within the complexity bound for $\mathcal{C}$. It is a standard fact that $\mathsf{LOGSPACE}^{\mathsf{LOGSPACE}} = \mathsf{LOGSPACE}$ and that $\mathsf{P}^{\mathsf{P}} = \mathsf{P}$. For precise definitions see also some textbook, e. g. [Sip96]. Note that our definition of oracle classes is slightly non-standard by allowing more than one oracle from a finite fixed set.

Circuit Complexity. On the lowest levels in the complexity hierarchy, we apply circuit complexity. The classes NC^i (resp. AC^i, resp. TC^i) are defined as the classes of languages accepted by a uniform family of circuits of depth $\log^i N$ (where N is the

input length) with bounded fan-in Boolean gates (and $\wedge$, or $\vee$, not $\neg$) (resp. unbounded fan-in Boolean gates, resp. unbounded fan-in Boolean and MAJORITY gates). Here, the *depth* of a circuit is the length of the longest path from some input gate to an output gate. For more details on these definitions, see Appendix A. As MAJORITY gates can be simulated by NC^1 circuits, we have the hierarchy

$$\mathsf{AC}^0 \subseteq \mathsf{TC}^0 \subseteq \mathsf{NC}^1 \subseteq \mathsf{AC}^1 \subseteq \mathsf{TC}^1 \subseteq \mathsf{NC}^2 \subseteq \cdots \subseteq \bigcup_{i\in\mathbb{N}} \mathsf{NC}^i = \mathsf{NC} \subseteq \mathsf{P}.$$

Moreover, we have $\mathsf{NC}^1 \subseteq \mathsf{LOGSPACE} \subseteq \mathsf{AC}^1$.

Reductions with Circuits. Let K, L be languages. We say K is AC^0-*reducible* to L if there is an AC^0 circuit deciding K which in addition to the Boolean gates also may use oracle gates for L (i. e., gates which output 1 if and only if the word encoded by the input of the gate is in L). It is straightforward to see that AC^0 reducibility is transitive and that if K is AC^0 reducible to L and $L \in \mathsf{AC}^0$, then also $K \in \mathsf{AC}^0$.

When talking about complete (resp. hard) problems for classes inside $\mathsf{LOGSPACE}$ (e. g. TC^0), we always assume that they are complete (resp. hard) with respect to AC^0 reductions.

We can also use oracle gates in NC^i circuits: Let $\mathcal{C}$ be some class of languages. We define $\mathsf{NC}^i(\mathcal{C})$ to be the class of languages decided by a family of NC^i circuits which may also use oracle gates for a finite subset of languages in $\mathcal{C}$. Here, the depth requirement of NC^i is modified that way that a gate with fan-in k for some language $L' \in \mathcal{L}$ counts as depth $\log k$. With this definition, we have $\mathsf{NC}^j(\mathsf{NC}^i) = \mathsf{NC}^{j+i-1}$ and $\mathsf{NC}^j(\mathsf{AC}^i) = \mathsf{AC}^{j+i-1}$ for all $j, i \geq 1$, see [Vol99, Thm. 4.21, Cor. 4.24]. If $L \in \mathsf{NC}^1(L')$, then L is called NC^1-(Turing-)reducible to L'.

The Class TC^0 and Arithmetic. Of particular interest for us is the class TC^0, which is contained in $\mathsf{LOGSPACE}$. Although it is a very low parallel complexity class, it is still very powerful with respect to arithmetic. By the very definition of AC^0 reducibility, we have:

Remark 2.3. MAJORITY is TC^0-complete.

ITERATED ADDITION (resp. ITERATED MULTIPLICATION) are the following computation problems: On input of n binary integers $a_1, \ldots, a_n$ each having n bits (i. e., the input length is $N = n^2$), compute the binary representation of the sum $\sum_{i=0}^{n} a_i$ (resp. product $\prod_{i=0}^{n} a_i$). For INTEGER DIVISION, the input are two binary n-bit integers a, b; the binary representation of the integer $c = \lfloor a/b \rfloor$ has to be computed. The first statement of the following theorem is a standard fact, see [Vol99]; the other statements are due to Hesse, [Hes01, HAB02].

Theorem 2.4 ([Hes01, HAB02, Vol99])**.** *The following problems are all in uniform* TC^0*:*

- ITERATED ADDITION,

- Iterated Multiplication,
- Integer Division.

The Class LOGDCFL. The classes LOGDCFL (resp. LOGCFL) are the classes of languages which are LOGSPACE-reducible to a deterministic context-free language (resp. to a context-free language). Trivially, we have $\mathsf{LOGSPACE} \subseteq \mathsf{LOGDCFL} \subseteq \mathsf{LOGCFL}$. In [Ven91], Venkateswaran showed that $\mathsf{LOGCFL} = \mathsf{SAC}^1$ (i. e., AC^1 where $\wedge$ gates are restricted to fan-in two and $\neg$ gates may only appear directly after input gates). This gives the connection to circuit complexity, again.

The results of Theorem 2.5 are stated in [Sud77] without a proof. The proof is straightforward using Theorem 2.6 below. In [GLS02], a proof of the analog statement for LOGCFL can be found (which is more tricky).

Theorem 2.5 ([Sud77, GLS02])**.** *We have* $\mathsf{LOGSPACE}^{\mathsf{LOGDCFL}} = \mathsf{LOGDCFL}$*. Moreover,* LOGDCFL *is closed under* $\mathsf{LOGSPACE}^{\mathsf{LOGDCFL}}$ *reductions.*

A DLPAuxPDA is a deterministic LOGSPACE-Turing machine which additionally has access to a stack and runs in polynomial time. A stack is an additional work tape without space restriction where the read-write-head stands always on the rightmost non-blank symbol – every time it moves right, it writes a new symbol on the tape; every time it moves left, it deletes the last symbol. We denote the class of languages recognized by a DLPAuxPDA by L(DLPAuxPDA). Note that with this model of computation we have to require explicitly that the running time is polynomial – this does not follow automatically from the logarithmic space bound as the size of the stack is not limited.

Theorem 2.6 ([Sud77, Sud78])**.**

$$L(\text{DLPAuxPDA}) = \mathsf{LOGDCFL}.$$

2.6. Generic Complexity

For many practical applications, the "generic-case behavior" of an algorithm is more important than its average-case or worst-case behavior. The foundations of the theory of generic complexity were developed by Kapovich, Myasnikov, Schupp and Shpilrain in [KMSS03, KMSS05]. In [MSU08], applications in cryptography can be found. The notion of *generic complexity* refers to partial algorithms which are defined on (strongly) generic sets $I \subseteq D$. As before, $D = \bigcup_{N\in\mathbb{N}} D^{(N)}$ is some domain and all $D^{(N)}$ are finite. A set $I \subseteq D$ is called *generic* if

$$\lim_{N\to\infty} \frac{\left|D^{(N)} \setminus I\right|}{|D^{(N)}|} = 0.$$

Furthermore, I is called *strongly generic*, (sometimes also *exponentially generic*, see [MRF12]) if there exists some $\varepsilon > 0$ such that

$$\lim_{N\to\infty} 2^{\varepsilon N} \cdot \frac{\left|D^{(N)} \setminus I\right|}{|D^{(N)}|} = 0.$$

This means that the probability to find a random string outside I converges exponentially fast to zero.

Let $\mathcal{C}$ be some complexity class. A formal language or problem is said to be (strongly) generically in $\mathcal{C}$ if there is an algorithm $\mathcal{A}$ and a (strongly) generic set $I \subseteq D$ such that the algorithm $\mathcal{A}$ computes the correct solution within the resource bounds of $\mathcal{C}$ for inputs of I. For inputs outside of I, the algorithm $\mathcal{A}$ does not need to terminate; however, if it terminates, the answer has to be correct. Thus, if some algorithm runs in polynomial time on a strongly generic set, then, for practical purposes, it behaves as a polynomial time (worst case) algorithm. This is true although the average-case complexity of the algorithm can be arbitrarily high – as it is the case for our algorithm for the conjugacy problem of the Baumslag group, see Chapter 8.

For the conjugacy problem, the input consists of a pair of words $(v, w) \in \Sigma^* \times \Sigma^*$. According to our definition (which is as in [KMSS03]) a set $I \subseteq \Sigma^* \times \Sigma^*$ is generic if and only if $\lim_{N\to\infty} \frac{|D^{(N)}\setminus I|}{|D^{(N)}|} = 0$ where $D^{(N)} = \{(v, w) \in \Sigma^* \times \Sigma^* \mid |v| + |w| \leq N\}$. This definition leads to the following theorem:

Theorem 2.7 ([KMSS03, Thm. C])**.** *Let G be some finitely generated group and $G \to H$ be a surjective homomorphism onto some infinite abelian group. Then the conjugacy problem of G is generically in linear time.*

Note that on a generic set of inputs the algorithm for Theorem 2.7 can detect that two elements are not conjugate (in some cases), but it never gives a positive answer. On the other hand, the requirements of Theorem 2.7 are really weak: it is even not required that the word problem is decidable.

In many cases we can show stronger results by imposing stronger requirements. In particular, often there is a (strongly) generic set $I \subseteq \Sigma^*$ such that the conjugacy problem can be decided as soon as one of the input words is in I. This leads to the following definition: Let $D = \widetilde{D} \times \widetilde{D}$. A set $I \subseteq D$ is called *doubly (strongly) generic* if there is a (strongly) generic set $\tilde{I} \subseteq \widetilde{D}$ with $\tilde{I} \times \tilde{I} \subseteq I$. A formal language or problem for which the input consists of a pair in $\widetilde{D} \times \widetilde{D}$, is called *doubly (strongly) generically* in some complexity class $\mathcal{C}$ if there is a doubly (strongly) generic set I and an algorithm which on inputs of I, computes the correct solution within the bounds of $\mathcal{C}$.

The next result shows that in our case, in fact, "doubly (strongly) generic" implies "(strongly) generic".

Proposition 2.8. *Let $\widetilde{D} = \Sigma^*$ and $D = \widetilde{D} \times \widetilde{D}$ with $\widetilde{D}^{(N)} = \{w \in \Sigma^* \mid |w| \leq N\}$ and $D^{(N)} = \{(v, w) \in \Sigma^* \times \Sigma^* \mid |v| + |w| \leq N\}$. If $I \subseteq D$ is doubly (strongly) generic, then I is also (strongly) generic.*

Proof. Let $\tilde{I}$ be a generic set such that $\tilde{I} \times \tilde{I} \subseteq I$. Let $\delta > 0$. For N large enough, we have

$$\sum_{k=0}^{\lfloor N/4 \rfloor} \sum_{i=0}^{k} \left|\Sigma^i\right| \cdot \left|\Sigma^{k-i}\right| \leq \frac{N^2}{4^2} \left|\Sigma^{N/4}\right|^2 \leq \delta \cdot \left|\Sigma^N\right|,$$

and as $\tilde{I}$ is generic, also $\left|\Sigma^k \setminus \tilde{I}\right| \leq \delta \cdot \left|\Sigma^k\right|$ for $N/8 \leq k \leq N$. Thus, for large enough N, we obtain

$$\begin{aligned}
\left|D^{(N)} \setminus I\right| &= \sum_{k=0}^{N} \sum_{i=0}^{k} \left|\left(\Sigma^i \times \Sigma^{k-i}\right) \setminus I\right| \\
&\leq \sum_{k=0}^{N} \sum_{i=0}^{k} \left|\left(\Sigma^i \setminus \tilde{I}\right)\right| \cdot \left|\left(\Sigma^{k-i} \setminus \tilde{I}\right)\right| \\
&\leq \delta \cdot \left(\left|\Sigma^N\right| + \sum_{k=N/4}^{N} \sum_{i=0}^{k} \left|\Sigma^i\right| \cdot \left|\Sigma^{k-i}\right|\right) \\
&\leq 2\delta \left|D^{(N)}\right|.
\end{aligned}$$

Hence, $\left|D^{(N)} \setminus I\right| \in o\left(\left|D^{(N)}\right|\right)$. Analogously the result for strongly generic follows. □

Chapter 3.

Graphs of Groups and the Conjugacy Criterion

In this section, we give a short introduction to Bass-Serre theory. Our presentation is a shortened version taken from [DW13]. Many of the results will be used later – some without being referenced explicitly. Our focus is on rewriting techniques for fundamental groups of graphs of groups; we only give a slight glimpse on the other aspects of Bass-Serre theory. The results presented here are from [Ser80] (English translation of [Ser77]) and also our notation follows this book. However, we differ from [Ser80] by using rewriting techniques as presented in Section 2.1 and 2.2. We start with a general introduction to Bass-Serre theory – without giving proofs. Then we focus on Britton reductions. Finally, we examine conjugacy in fundamental groups of graphs of groups more carefully leading to the main tool for all our further results – the Conjugacy Criterion, Theorem 3.19.

3.1. Bass-Serre Theory

Definition 3.1 (Graph of Groups)**.** Let $Y = (V(Y), E(Y))$ be a connected graph. A *graph of groups* $\mathcal{G}$ over Y is given by the following data:

(i) For each vertex $P \in V(Y)$, there is a *vertex group* G_P.

(ii) For each edge $y \in E(Y)$, there is an *edge group* G_y such that $G_y = G_{\overline{y}}$.

(iii) For each edge $y \in E(Y)$, there is an injective homomorphism from G_y to $G_{\iota(y)}$, which is denoted by $a \mapsto a^y$. The image of G_y in $G_{\iota(y)}$ is denoted by G_y^y.

Since we have $G_y = G_{\overline{y}}$, there is also a homomorphism $G_y \to G_{\tau(y)}$ with $a \mapsto a^{\overline{y}}$. The image of G_y in $G_{\tau(y)}$ is denoted by $G_y^{\overline{y}}$. Thus, for $y \in E(Y)$ with $\iota(y) = P$ and $\tau(y) = Q$, there are two isomorphisms and inclusions:

$$\begin{aligned} G_y \tilde{\to} G_y^y \leq G_P, \qquad & a \mapsto a^y, \\ G_y \tilde{\to} G_y^{\overline{y}} \leq G_Q, \qquad & a \mapsto a^{\overline{y}}. \end{aligned}$$

The graph of groups is called finite if Y is a finite graph.

In the following, we only consider finite graphs of groups.

Example 3.2. Graphs of groups arise in a natural way in situations where a group G acts on some tree $X = (V, E)$ without edge inversion. We let $Y = G\backslash X$ be the quotient graph with vertex set $V(Y) = \{Gv \mid v \in V\}$ and edge set $E(Y) = \{Ge \mid e \in E\}$. Choosing representatives, we may assume $V(Y) \subseteq V$ and $E(Y) \subseteq E$. For $P \in V(Y)$, $y \in E(Y)$, we define vertex and edge groups as the stabilizers of the respective representatives: $G_P = \{g \in G \mid gP = P\}$ and $G_y = \{g \in G \mid gy = y\}$. Now, for each $y \in E(Y)$, there are $P, Q \in V(Y)$ and $g_y, h_y \in G$ such that $\iota(y) = g_y P$ and $\tau(y) = h_y Q$. This yields two embeddings:

$$\begin{aligned} G_y &\xrightarrow{\sim} G_y^y \leq G_P \leq G, \qquad a \mapsto a^y = \overline{g}_y a g_y, \\ G_y &\xrightarrow{\sim} G_y^{\overline{y}} \leq G_Q \leq G, \qquad a \mapsto a^{\overline{y}} = \overline{h}_y a h_y. \end{aligned}$$

As G_y and $G_{\overline{y}}$ are isomorphic, we have obtained a well-defined graph of groups over Y.

For developing the fundamental group, we begin with the group $F(\mathcal{G})$. As a symmetric set of generators (as monoid) we choose a disjoint union

$$\Delta = \bigcup_{P \in V(Y)} (G_P \setminus \{1\}) \cup E(Y).$$

Note that this might be an infinite alphabet. We fix the alphabet Δ throughout this chapter – and also in some of the following chapters. Now, we have

$$F(\mathcal{G}) = F_\Delta \Big/ \left\{ gh = [gh],\ y a^{\overline{y}} \overline{y} = a^y \;\middle|\; P \in V(Y),\ g, h \in G_P;\ y \in E(Y),\ a \in G_y \right\},$$

where as usual F_Δ is the free group with basis Δ and $[gh]$ denotes the element obtained by multiplying g and h in G_P (where $1 \in G_P$ is identified with the empty word).

For each edge $y \in E(Y)$ with $\iota(y) = P$, we choose a set C_y of representatives of the left cosets G_P / G_y^y with $1 \in C_y$. Thus, each $g \in G_P$ admits a unique factorization $g = ca^y$ with $c \in C_y$ and $a \in G_y$. We now define a rewriting system $S_\mathcal{G} \subseteq \Delta^* \times \Delta^*$:

$$\begin{array}{rcll} gh & \longrightarrow & [gh] & \text{for } P \in V(Y), g, h \in G_P \setminus \{1\}, \\ [ca^y]y & \longrightarrow & cya^{\overline{y}} & \text{for } y \in E(Y), c \in C_y, a \in G_y \setminus \{1\}, \\ \overline{y}y & \longrightarrow & 1 & \text{for } y \in E(Y). \end{array}$$

Proposition 3.3. *We have $F(\mathcal{G}) = \Delta^*/S_\mathcal{G}$, and $S_\mathcal{G}$ is a convergent semi-Thue system.*

Let us define subsets of Δ^* as follows: For $P, Q \in V(Y)$, we denote with $\Pi(\mathcal{G}, P, Q)$ the set of words where the occurring edges form a path from P to Q in Y and the group elements of vertex groups between two edges are always of the corresponding vertex in the path; more precisely,

$$\begin{aligned} \Pi(\mathcal{G}, P, Q) = \Big\{ g_0 y_1 \cdots g_{n-1} y_n g_n \;\Big|\; & y_i \in E(Y),\ \iota(y_1) = P,\ \tau(y_n) = Q, \\ & \tau(y_i) = \iota(y_{i+1}),\ g_0 \in G_P,\ g_i \in G_{\tau(y_i)} \text{ for all } i \Big\}, \end{aligned}$$

and we set

$$\Pi(\mathcal{G}) = \bigcup_{P \in V(Y)} \Pi(\mathcal{G}, P, P).$$

Note that the image of $\Pi(\mathcal{G})$ in $F(\mathcal{G})$ is not a group but a so-called *groupoid*. Elements in $\Pi(\mathcal{G}) \cap \mathrm{IRR}(S_{\mathcal{G}})$ have the form $c_0 y_1 \cdots c_{n-1} y_n g_n$ with $c_i \in C_{y_{i+1}}$ and $g_n \in G_{\tau(y_n)}$.

If $w = g_0 y_1 \cdots g_{n-1} y_n g_n \in \Pi(\mathcal{G})$, then we call w a *$\mathcal{G}$-factorization* of the respective group element in $F(\mathcal{G})$; by saying this we implicitly require that $y_i \in E(Y), \tau(y_i) = \iota(y_{i+1})$, $g_0 \in G_{\iota(y_1)}$, $g_i \in G_{\tau(y_i)}$ for all i. The *$\mathcal{G}$-length* $|w|_{\mathcal{G}}$ of some word $w = g_0 y_1 \cdots g_{n-1} y_n g_n \in \Pi(\mathcal{G}, P, Q)$ for $P, Q \in V(Y)$ is n – i. e., it is the number of occurring edges. In particular, if $w = g_0 \in G_P$ for some $P \in V(\Gamma)$, then $|w|_{\mathcal{G}} = 0$. Moreover, we call $y_1 \cdots y_n$ the *underlying path* or *edge sequence* of w.

If Σ_P are (monoid) generating sets of G_P for $P \in V(Y)$, and

$$\Sigma = \bigcup_{P \in V(Y)} \Sigma_P \cup E(Y), \tag{3.1}$$

then we also call a word $w = g_0 y_1 \cdots g_{n-1} y_n g_n \in \Sigma^*$ with $g_0 \in \Sigma^*_{\iota(y_1)}$ and $g_i \in \Sigma^*_{\tau(y_i)}$ a $\mathcal{G}$-factorization if the projection onto Δ^* (i. e., if the words g_i are read as group elements of the vertex groups) is a $\mathcal{G}$-factorization. Also the $\mathcal{G}$-length of w is defined as the $\mathcal{G}$-length of the projection. Throughout (with few exceptions), we use the small letter n for the $\mathcal{G}$-length, whereas the capital letter N denotes the length of the word over Σ.

Note that for all vertices $P \in V(Y)$, the image of $\Pi(\mathcal{G}, P, P)$ in $F(\mathcal{G})$ is a group.

Definition 3.4. Let $P \in V(Y)$. The *fundamental group* $\pi_1(\mathcal{G}, P)$ of $\mathcal{G}$ with respect to the base point $P \in V(Y)$ is defined as the image of $\Pi(\mathcal{G}, P, P)$ in $F(\mathcal{G})$.

Recall that we have made the hypothesis that Y is connected. Thus, there exists a spanning tree $T = (V(Y), E(T))$ of Y.

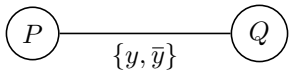
Definition 3.5. Let T be a spanning tree of Y. The *fundamental group* of $\mathcal{G}$ with respect to T is defined by:

$$\pi_1(\mathcal{G}, T) = F(\mathcal{G}) \Big/ \{y = 1 \mid y \in T\}.$$

Proposition 3.6. *The canonical homomorphism from the subgroup $\pi_1(\mathcal{G}, P)$ of $F(\mathcal{G})$ to the quotient group $\pi_1(\mathcal{G}, T)$ is an isomorphism.*

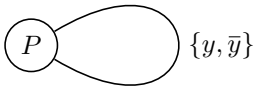
Proposition 3.6 shows, in particular, that the two definitions of a fundamental group are independent of the choice of the base point or the spanning tree. As an abstract group, we denote it by $\pi_1(\mathcal{G})$. It is the fundamental group of the graph of groups $\mathcal{G}$ over the connected graph Y. Note that for every $P \in V(Y)$ the canonical homomorphism $G_P \to \pi_1(\mathcal{G}, T)$ is injective.

Example 3.7. Let G_P and G_Q be two groups with a common subgroup $G_y = G_P \cap G_Q$. Then the *amalgamated product* $G_P *_{G_y} G_Q$ of G_P and G_Q over G_y is the fundamental group of the graph of groups over Y, where Y is as follows:



Sometimes we also write $G_P *_{G_y^y = G_{\overline{y}}^{\overline{y}}} G_Q$. The group G_y is called amalgamated subgroup. In the case that G_y is the trivial group, we obtain the *free product* $G_P * G_Q$.

Example 3.8. Let $\mathcal{G}$ be a graph of groups over the following graph:



Then the fundamental group $\pi_1(\mathcal{G})$ is the *HNN extension* (Higman, Neumann, Neumann, [HNN49]) with *stable letter* y which is defined as

$$\pi_1(\mathcal{G}) = \left(G_P * F_{\{y\}}\right) \Big/ \left\{ y a^{\overline{y}} \overline{y} = a^y \;\middle|\; a \in G_y \right\}.$$

We also write $\langle\, G_P, y \mid yay^{-1} = \varphi(a) \text{ for } a \in \mathrm{dom}(\varphi) \,\rangle$ if $\varphi : G_{\overline{y}}^{\overline{y}} \to G_y^y$ is the isomorphism defined by the inclusions of the edge group.

The group G_y is also called *associated subgroup* and G_P is called the *base group*. If the graph Y consists of one vertex with k undirected edges, the fundamental group is called an HNN extension of the vertex group with k stable letters. As it is common in literature, we also use the letter t instead of y as stable letter for HNN extensions.

Definition 3.9. A graph of groups is called *reduced* if for all edges y with $\iota(y) \neq \tau(y)$ (i. e., which are not loops) the inclusion $G_y \to G_{\iota(y)}$ is *not* surjective.

Proposition 3.10. *For every graph of groups $\mathcal{G}$, there is a reduced graph of groups $\mathcal{G}'$ such that $\pi_1(\mathcal{G}') = \pi_1(\mathcal{G})$.*

Proposition 3.11. *Let $\mathcal{G}$ be a graph of groups and let $\mathcal{H}$ be another graph of groups with the same underlying graph Y and let $H_y = G_y$ for all $y \in E(Y)$ and $H_P \leq G_P$ for all $P \in V(Y)$; moreover, let also be the embeddings $G_y \to G_{\iota(y)}$ and $H_y \to H_{\iota(y)}$ coincide. Then there is a canonical injective homomorphism $\pi_1(\mathcal{H}) \to \pi_1(\mathcal{G})$.*

The following result is the main theorem of Bass-Serre theory.

Theorem 3.12 ([Ser80, Thm. 13])**.** *Let X be a tree and let a group G act without edge inversion on X. Let $\mathcal{G}$ be the associated graph of groups over $Y = G\backslash X$ as in Example* 3.2*. Then there is an isomorphism $\pi_1(\mathcal{G}) \to G$.*

3.2. Britton Reductions over Graphs of Groups

For some purposes, we do not need unique normal forms; hence, the rewriting system $S_{\mathcal{G}}$ is too complicated. The following observation gives rise to another rewriting system: If $w = g_0 y_1 \cdots g_{n-1} y_n g_n \in \Pi(\mathcal{G}, P, Q)$, and $y_i g_i y_{i+1} = y a^{\overline{y}} \overline{y}$ with $a \in G_y$ for some i (this implies $y a^{\overline{y}} \overline{y} \overset{*}{\underset{S_{\mathcal{G}}}{\Longrightarrow}} a^y$), then also

$$g_0 y_1 \cdots g_{i-2} y_{i-1} [g_{i-1} a^y g_{i+1}] y_{i+2} g_{i+2} \cdots y_n g_n \in \Pi(\mathcal{G}, P, Q).$$

This leads to the notion of *Britton reduction* (see also [LS01, Sec. IV.2]). In [Bri63], they were originally defined for HNN extensions. Britton reductions are given by the rewriting system $B_{\mathcal{G}} \subseteq \Delta^* \times \Delta^*$ with the following rules:

$$\begin{aligned} gh &\longrightarrow [gh] && \text{for } P \in V(Y),\ g, h \in G_P \setminus \{1\}, \\ y a^{\overline{y}} \overline{y} &\longrightarrow a^y && \text{for } y \in E(Y),\ a \in G_y. \end{aligned}$$

As $B_{\mathcal{G}}$ is length-reducing, it is terminating. Furthermore, the reflexive, symmetric and transitive closures $\overset{*}{\underset{B_{\mathcal{G}}}{\Longleftrightarrow}}$ and $\overset{*}{\underset{S_{\mathcal{G}}}{\Longleftrightarrow}}$ are the same. In particular, $F(\mathcal{G}) = \Delta^* / B_{\mathcal{G}}$.

A word $w \in \Delta^*$ is called *Britton-reduced* if $w \in \mathrm{IRR}(B_{\mathcal{G}})$. If Σ is a generating set as in (3.1) and $v \in \Sigma^*_{\tau(y)}$, $w \in \Sigma^*_{\iota(y)}$, we call a rewriting step $yv\overline{y} \to w$ a *Britton reduction* if $v =_{G_{\tau(y)}} a^{\overline{y}}$ and $w =_{G_{\iota(y)}} a^y$ for some $a \in G_y$. In particular, a $\mathcal{G}$-factorization $w \in \Sigma^*$ is called Britton reduced if there is no factor $yv\overline{y}$ with $v \in_{G_{\tau(y)}} G_y^{\overline{y}}$. (Be aware of the slight ambiguity in the case of $\Sigma = \Delta$.) The following facts are crucial:

Lemma 3.13.

(i) $\mathrm{IRR}(S_{\mathcal{G}}) \subseteq \mathrm{IRR}(B_{\mathcal{G}})$.

(ii) *Let $w \in \Pi(\mathcal{G}, P, Q)$ and $w =_{F(\mathcal{G})} \tilde{w}$ for some $\tilde{w} \in \mathrm{IRR}(B_{\mathcal{G}})$. Then $\tilde{w} \in \Pi(\mathcal{G}, P, Q)$, and the underlying path of $\tilde{w}$ is uniquely defined. If furthermore $w \overset{*}{\underset{S_{\mathcal{G}}}{\Longrightarrow}} \hat{w} \in \mathrm{IRR}(S_{\mathcal{G}})$, then also $\hat{w} \in \Pi(\mathcal{G}, P, Q)$, and the underlying paths of $\tilde{w}$ and $\hat{w}$ coincide.*

(iii) *If $w = c_0 y_1 \cdots c_{n-1} y_n g_n \in \mathrm{IRR}(S_{\mathcal{G}}) \cap \Pi(\mathcal{G}, P, Q)$, then the prefix $c_0 y_1 \cdots c_{n-1} y_n$ depends on $wG_Q \subseteq F(\mathcal{G})$, only.*

(iv) *If $v, w \in \mathrm{IRR}(B_{\mathcal{G}}) \cap \Pi(\mathcal{G})$ with $v =_{F(\mathcal{G})} w$, then v can be transformed into w by only applying rules of the form*

$$[g a^y] y h \longrightarrow g y [a^{\overline{y}} h] \qquad \text{for } y \in E(Y), a \in G_y, g \in G_{\iota(y)}, h \in G_{\tau(y)}.$$

Note that in contrast to $S_{\mathcal{G}}$ these rules are symmetric.

As an immediate consequence, it follows:

Lemma 3.14 (Britton's Lemma, [Bri63])**.** *Let $w \in \mathrm{IRR}(B_{\mathcal{G}})$ with $w =_{F(\mathcal{G})} 1$. Then $w = 1$ (i. e., w is the empty word).*

Lemma 3.14 gives us an easy algorithm to decide the word problem of $\pi_1(\mathcal{G})$.

Lemma 3.15. *Let $\mathcal{G}$ be a graph of groups with underlying graph Y. For all $y \in E(Y)$, let the subgroup membership problem of G_y^y in $G_{\iota(y)}$ be decidable and let the word problem of G_P be decidable for some $P \in V(Y)$. Furthermore, let the isomorphisms $G_y^y \to G_y^{\bar{y}}$ be effectively computable for all $y \in E(Y)$. Then there is an algorithm which on input $w \in \Pi(\mathcal{G})$, computes a Britton-reduced word $\hat{w} \in \Pi(\mathcal{G})$ with $\hat{w} =_{F(\mathcal{G})} w$. In particular, the word problem of $\pi_1(\mathcal{G})$ is decidable.*

Proof. We consider the fundamental group $\pi_1(\mathcal{G}, P)$ with respect to some base point P. Britton reductions can be effectively computed since the subgroup membership problem for G_y^y in $G_{\iota(y)}$ is decidable and the isomorphisms $G_y^y \to G_y^{\bar{y}}$ are computable. Hence, on input $w \in \Pi(\mathcal{G})$, one only needs to apply Britton reductions until it is not possible anymore. If at the end there is still some y in the resulting word, the input word is not equal to 1 in $\pi_1(\mathcal{G}, P)$ by Lemma 3.14. Otherwise, the algorithm for the word problem of G_P is applied. □

Note that Lemma 3.15 does not state anything about the complexity. The problem is that – even if all computations can be performed efficiently – the blow up due to the calculations of the isomorphisms $G_y^y \to G_y^{\bar{y}}$ might prevent an efficient solution of the word problem in $\pi_1(\mathcal{G})$. An example is the Baumslag group $\mathbf{G}_{1,2} = \langle\, a, t, b \mid tat^{-1} = a^2, bab^{-1} = t \,\rangle$ which is an HNN extension of the Baumslag-Solitar group $\mathbf{BS}_{1,2}$. For $\mathbf{G}_{1,2}$, the straightforward algorithm, as in Lemma 3.15, leads to a non-elementary running time. However, in [MUW11] it is shown that the word problem still can be solved in polynomial time, see Chapter 8.

3.3. Conjugacy and Graphs of Groups

If $g, h \in G$ are conjugate, we write $g \sim h$. If the ambient group G is not clear from the context, we also write $g \sim_G h$ or "$g \sim h$ in G" if there is some $z \in G$ with $zgz^{-1} = h$. Note that this is even defined if g and h are not in G.

Lemma 3.16. *Let $v \in \Pi(\mathcal{G}, P, P)$, $w \in \Pi(\mathcal{G}, Q, Q)$ for some $P, Q \in V(Y)$. If $u \in \mathrm{IRR}(B_{\mathcal{G}})$ with $uw\bar{u} =_{F(\mathcal{G})} v$, then $u \in \Pi(\mathcal{G}, P, Q)$.*

Proof. Because of Lemma 3.13 (ii), we may assume that v and w are Britton-reduced. If $uw\bar{u}$ is also Britton-reduced, then $u \in \Pi(\mathcal{G}, P, Q)$ by Lemma 3.13 (ii), and we are done. Otherwise, we can write $u = u'y$ for some $y \in E(Y)$ or $u = u'g$ for some $g \in G_R$ with $R \in V(Y)$. Now, in $uw\bar{u}$ there has to be a place where a Britton reduction can be applied. By assumption, a Britton reduction has to involve letters from both w and u. Hence, in particular, we have $\tau(y) = Q$ resp. $g \in G_Q$ (i. e., $R = Q$). Thus, the lemma follows by induction. □

Corollary 3.17. *Let $g, h \in \pi_1(\mathcal{G}, P)$. If $g \sim h$ in $F(\mathcal{G})$, then also $g \sim h$ in $\pi_1(\mathcal{G}, P)$.*

By Corollary 3.17, instead of testing conjugacy in the fundamental group $\pi_1(\mathcal{G}, P)$, we can test it in the larger group $F(\mathcal{G})$. This simplifies the algorithms substantially because for $\mathcal{G}$-factorizations in $F(\mathcal{G})$ there is good notion of cyclically reduced elements.

Let $w = g_0 y_1 g_1 \cdots y_n g_n \in \Pi(\mathcal{G})$. We say that v is a *cyclic permutation* or *transposition* of w if there are $u, u' \in \Delta^*$ such that $w = uu'$ and $v = u'u$. A word $w \in \Delta^*$ is called *cyclically Britton-reduced* or simply *cyclically reduced* if every cyclic permutation of w is Britton-reduced. That means w is cyclically reduced if and only if ww is Britton-reduced or $|w|_{\mathcal{G}} = 0$.

Lemma 3.18. *Let $w = y_1 g_1 \cdots y_n g_n \in \Pi(\mathcal{G}) \cap \mathrm{IRR}(B_{\mathcal{G}})$ with $n \geq 1$. Then one of the following two cases holds:*

(i) *There is some $m \in \mathbb{N}$ with $m + 1 \leq n - m$ and some $\tilde{g} \in G_{\tau(y_{n-m})}$ and*

$$ww = y_1 g_1 \cdots y_n g_n y_1 g_1 \cdots y_n g_n \overset{*}{\underset{B_{\mathcal{G}}}{\Longrightarrow}} y_1 g_1 \cdots y_{n-m} \tilde{g} y_{m+1} g_{m+1} \cdots y_n g_n \in \mathrm{IRR}(\mathcal{B}_{\mathcal{G}}).$$

In this case, $\hat{w} = y_{m+1} g_{m+1} \cdots y_{n-m} \tilde{g}$ is cyclically reduced and $w \sim_{F(\mathcal{G})} \hat{w}$.

(ii) *Otherwise, n is even and*

$$y_{n/2+1} g_{n/2+1} \cdots y_n g_n \cdot y_1 g_1 \cdots y_{n/2} g_{n/2} \overset{*}{\underset{B_{\mathcal{G}}}{\Longrightarrow}} \tilde{g} \in G_{\tau(y_{n/2})}$$

and $\tilde{g} \sim_{F(\mathcal{G})} w$ is cyclically reduced.

Lemma 3.18 is the basic tool to reduce $\mathcal{G}$-factorizations cyclically. As every $\mathcal{G}$-factorization is in $\pi_1(\mathcal{G}, P)$ for some $P \in V(Y)$, it suffices to apply Britton reductions to elements of $\pi_1(\mathcal{G}, P)$. Hence, although we work in the larger group $F(\mathcal{G})$, we only need to solve the word problem of $\pi_1(\mathcal{G}, P)$ to reduce $\mathcal{G}$-factorizations cyclically – and to solve conjugacy in case (iii) of Theorem 3.19 below.

Proof. First, we have to show that no other case can occur. Assume we are not in case (i), i. e., $y_1 g_1 \cdots y_{n-m} \cdot \tilde{g} \cdot y_{m+1} g_{m+1} \cdots y_n g_n \notin \mathrm{IRR}(\mathcal{B}_{\mathcal{G}})$ for $m = (n-2)/2$ (if n is even) resp. $m = (n-1)/2$ (if n is odd). If n is odd, then $y_{n-(n-1)/2} \cdot \tilde{g} \cdot y_{(n-1)/2+1} = y_{(n+1)/2} \cdot \tilde{g} \cdot y_{(n+1)/2}$ is Britton-reduced. Since this is the only place where a Britton reduction could possibly take place, this case cannot occur and n has to be even. If n is even and $y_1 g_1 \cdots y_{n/2+1} \cdot \tilde{g} \cdot y_{n/2} g_{n/2} \cdots y_n g_n \notin \mathrm{IRR}(\mathcal{B}_{\mathcal{G}})$, then we are in case (ii).

In case (i), $\hat{w}$ is obviously cyclically reduced. Moreover,

$$y_1 g_1 \cdots y_m g_m \cdot \hat{w} \cdot (y_1 g_1 \cdots y_m g_m)^{-1} =_{F(\mathcal{G})} w.$$

In the second case, it is also immediate that $\tilde{g}$ is cyclically reduced and $\tilde{g} \sim_{F(\mathcal{G})} w$. □

Let $\mathcal{A}$ denote the union of all G_y^y. The following result is due to Horadam [Hor81]; it is our main tool for deciding the conjugacy problem in the following chapters. For amalgamated products, it first appeared in [MKS66]; the special case for HNN extensions is known as *Collins' Lemma* [Col69], see also [LS01, Thm. IV.2.5].

Theorem 3.19 (Conjugacy Criterion, [Hor81]). *Let $w \in \Pi(\mathcal{G})$ be cyclically reduced. Then one of the following three cases holds:*

(i) *There is some $P \in V(Y)$ with $w \in G_P$ and $w \sim_{F(\mathcal{G})} a$ for some $a \in \mathcal{A}$. In this case, there exists a sequence of elements $a = a_0, a_1, \ldots, a_m \in \mathcal{A}$ such that $a_m \sim_{G_P} w$ and for every i there is some $b_i \in \Delta$ with $a_i = b_i a_{i-1} \bar{b}_i$.*

(ii) *There is some $P \in V(Y)$ with $w \in G_P$ and w is not conjugate to any $a \in \mathcal{A}$. In this case, if $w \sim_{F_\mathcal{G}} v$ for some cyclically reduced v, then $v \in G_P$ and $v \sim_{G_P} w$.*

(iii) *We have $w \notin G_P$ for any $P \in V(Y)$. Then we may assume that w starts with a letter $y \in E(Y)$ and has the form:*

$$w = y_1 g_1 \cdots y_n g_n$$

with $n \geq 1$. If w is conjugate to a cyclically reduced $\mathcal{G}$-factorization v which starts with some letter $x \in E(Y)$, then there is some $1 \leq i \leq n$ and $a \in G_{y_i}^{y_i} \subseteq \mathcal{A}$ such that

$$v =_{F(\mathcal{G})} a\, y_i g_i \cdots y_n g_n\, y_1 g_1 \cdots y_{i-1} g_{i-1}\, \bar{a},$$

i. e., w can be transformed into v by a cyclic permutation followed by a conjugation with an element of $\mathcal{A}$.

Proof. First, let $w \in G_P$ and let $w = ua\bar{u}$ for some $a \in \mathcal{A} \cap G_Q$ with $Q \in V(Y)$ and $u \in \Delta^*$. We may assume that u is Britton-reduced; hence, by Lemma 3.16, $u = h_0 x_1 h_1 \cdots x_m h_m \in \Pi(\mathcal{G}, P, Q)$. Let $u' = h_i x_{i+1} h_{i+1} \cdots x_m h_m$ for some $i > 0$ and assume we had $u' w \bar{u}' \notin G_{\bar{x}_i}^{\bar{x}_i} \subseteq \mathcal{A}$. Then, however, $uw\bar{u}$ could not be reduced to a single letter by applying Britton reductions. Hence, $u' w \bar{u}' \in G_{\bar{x}_i}^{\bar{x}_i} \subseteq \mathcal{A}$, and thus, we also have $x_i u' w \bar{u}' \bar{x}_i \in \mathcal{A}$. Therefore, for every proper suffix u' of u, we have $u' a \bar{u}' \in \mathcal{A}$, and hence, we have finished case (i).

Now, let $w \in G_P$ not be conjugate to any element in $\mathcal{A}$ and let $uw\bar{u} =_{F(\mathcal{G})} v$ for some cyclically reduced $\mathcal{G}$-factorization v and some Britton-reduced word u. If $u \notin G_P$, i. e., by Lemma 3.16, $u = u' y h$ for some $u' \in \Delta^*$, $y \in E(Y)$, and $h \in G_P$, then $u' y [h w \bar{h}] \bar{y} \bar{u}' =_{F(\mathcal{G})} uw\bar{u}$ is Britton-reduced; thus, by Lemma 3.13, it cannot be equal in $F(\mathcal{G})$ to any cyclically reduced word. Hence, $w \sim_{G_P} v$.

Finally, let

$$w = y_1 g_1 \cdots y_n g_n \in \Pi(\mathcal{G}, P, P)$$

for some $n \geq 0$ and $v \in \Pi(\mathcal{G}, Q, Q)$ be cyclically reduced and let u Britton-reduced with $uw\bar{u} =_{F(\mathcal{G})} v$. Moreover, assume that v starts with some letter $x \in E(Y)$. By Lemma 3.16, we know that $u = h_0 x_1 h_1 \cdots x_m h_m \in \Pi(\mathcal{G}, Q, P)$. If $u = h_0 \in G_P$ (i. e., $m = 0$), then we have $h_0 \in G_{y_1}^{y_1} \subseteq \mathcal{A}$, for otherwise, we had $h_0 w \bar{h}_0 \neq_{F(\mathcal{G})} v$ as v starts with some letter $x \in E(Y)$. Now, let $m \geq 1$. Then we have

$$v =_{F(\mathcal{G})} h_0 x_1 h_1 \cdots x_m h_m \cdot y_1 g_1 \cdots y_n g_n \cdot \bar{h}_m \bar{x}_m \cdots \bar{h}_1 \bar{x}_1 \bar{h}_0.$$

Since v is cyclically reduced, a Britton reduction has to be applicable. Without loss of generality we may assume

$$h_{m-1}x_m h_m\, y_1 g_1 \overset{*}{\underset{B_{\mathcal{G}}}{\Longrightarrow}} \tilde{h} \in G_{\iota(x_m)}$$

and we obtain

$$\begin{aligned} v &=_{F(\mathcal{G})} h_0 x_1 h_1 \cdots x_{m-1}\, \tilde{h} \cdot y_2 g_2 \cdots y_n g_n \cdot \underbrace{\overline{h}_m \overline{x}_m \overline{h}_{m-1}\, \tilde{h}}_{=_{F(\mathcal{G})}\ y_1 g_1} \tilde{h}^{-1}\, \overline{x}_{m-1} \cdots \overline{h}_1 \overline{x}_1 \overline{h}_0 \\ &=_{F(\mathcal{G})} h_0 x_1 h_1 \cdots x_{m-1} \tilde{h} \cdot y_2 g_2 \cdots y_n g_n y_1 g_1 \cdot \tilde{h}^{-1} \overline{x}_{m-1} \cdots \overline{h}_1 \overline{x}_1 \overline{h}_0 \\ &= u' \cdot y_2 g_2 \cdots y_n g_n y_1 g_1 \cdot \overline{u}' \end{aligned}$$

for $u' = h_0 x_1 h_1 \cdots x_{m-1} \tilde{h}$. Now, $|u'|_{\mathcal{G}} < |u|_{\mathcal{G}}$; so we can apply induction and the theorem follows since the concatenation of two cyclic permutations is a cyclic permutation. $\square$

Chapter 4.

Amenability and Generic Complexity

In [KMSS03], amenability is an important concept when talking about strongly generic sets in groups. One definition of a group being non-amenable is that the probability

$$p^{(n)}(1,1) = \frac{|\{w \in \Sigma^n \mid w =_G 1\}|}{|\Sigma^n|}$$

of the simple random walk decreases exponentially in the length n. That means, if some algorithm works for all group elements except the identity, then it is a strongly generic algorithm. For the conjugacy problem, we can often design algorithms which work outside the vertex groups (case (iii) of the Conjugacy Criterion, Theorem 3.19). Therefore, we are interested in the amenability of Schreier graphs of the fundamental group with respect to some vertex group.

We start this chapter by introducing the concept of amenability, then we show that the Schreier graphs we are interested in, in fact, are non-amenable; at the end, we relate amenability more precisely to generic sets.

4.1. Amenability

This section is devoted to give an overview of amenability and its connection to random walks. It is mainly based on the paper [CSGdlH99].

Let $\Gamma = (V(\Gamma), E(\Gamma), \iota, \tau, \bar{\cdot})$ be a graph (undirected, locally finite, but it might have loops and multi-edges – for definitions see Section 2.4). For $k \in \mathbb{N}$, we define the k-th neighborhood of some set of vertices $U \subseteq V(\Gamma)$ as

$$\mathcal{N}^k(U) = \{v \in V(\Gamma) \mid \exists\, u \in U \,:\, d(u,v) \leq k\}\,,$$

where $d(u,v)$ is the distance in the graph Γ. Moreover, we set $\beta U = \mathcal{N}^1(U) \cap \mathcal{N}^1(\overline{U})$ where $\overline{U} = V(\Gamma) \setminus U$ denotes the complement of U.

Γ is said to satisfy a *strong isoperimetric inequality* if there exists some $\varepsilon > 0$ such that for every finite $U \subseteq V(\Gamma)$ we have

$$|\beta U| \geq \varepsilon\,|U|\,.$$

A similar condition is the so-called *doubling condition*: there exists some $k \in \mathbb{N}$ such that for every finite $U \subseteq V(\Gamma)$ we have

$$\left|\mathcal{N}^k(U)\right| \geq 2\,|U|\,.$$

Lemma 4.1. *Let Γ be a graph of bounded degree. Then Γ satisfies the doubling condition if and only if it satisfies a strong isoperimetric inequality.*

Proof. In fact, the strong isoperimetric inequality implies the doubling condition for arbitrary graphs: If $|\beta U| \geq \varepsilon |U|$ for every finite U, then we have $|\mathcal{N}^2(U) \setminus U| = |\beta(\mathcal{N}^1(U))| \geq \varepsilon |\mathcal{N}^1(U)| \geq \varepsilon |U|$. Hence, $|\mathcal{N}^2(U)| \geq (1+\varepsilon)|U|$ for every finite U. By choosing $k = 2\left\lceil \log_{1+\varepsilon}(2) \right\rceil$, we obtain $\left|\mathcal{N}^k(U)\right| \geq 2|U|$.

For the other direction, note that for every $v \in \mathcal{N}^k(U) \setminus U$, there is a path of length at most $k-1$ from v to some $u \in \beta U$. Let d be a bound on the degree of Γ. It follows that $2|U| \leq \left|\mathcal{N}^k(U)\right| \leq |U| + \sum_{i=0}^{k-1} d^i |\beta U| \leq |U| + d^k |\beta U|$; hence $|\beta U| \geq \frac{1}{d^k}|U|$. □

Random Walks. The *simple random walk* on some (directed) graph is as follows: It starts at some vertex, chooses an outgoing edge uniformly at random and goes to the target vertex of this edge, then it chooses the next edge and so on. With $p^{(n)}(u,v)$ we denote the probability that the simple random walk on Γ ends after n steps in v when starting in u. More formally, let $u, v \in V(\Gamma)$; then, $p^{(n)}(u,v)$ is defined by the following recurrence:

$$\begin{aligned} p^{(0)}(u,v) &= \begin{cases} 1 & \text{if } u = v \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \\ p^{(n+1)}(u,v) &= \sum_{\substack{e \in E(\Gamma) \\ \tau(e) = v}} \frac{p^{(n)}(u, \iota(e))}{|\{f \in E(\Gamma) \mid \iota(f) = \iota(e)\}|}. \end{aligned} \tag{4.1}$$

Hence, the simple random walk on Γ is a Markov chain with state space $V(\Gamma)$ – outgoing edges are transitions which all are chosen with equal probability. From now on, let Γ be a d-regular graph – possibly having loops and multi-edges. That means there are exactly d^n different paths of length n starting at some fixed vertex. Thus, we have

$$p^{(n)}(u,v) = \frac{\text{number of paths from } u \text{ to } v}{d^n}.$$

The Spectral Radius. Let $\ell^2(V(\Gamma))$ be the Hilbert space of functions $x : V(\Gamma) \to \mathbb{R}$ such that $\|x\|_2 = \sum_{v \in V(\Gamma)} x(v)^2 < \infty$ with the inner product $\langle x, y \rangle = \sum_{v \in V(\Gamma)} x(v) y(v)$ for $x, y \in \ell^2(V(\Gamma))$. For $x \in \ell^2(V(\Gamma))$, we denote the support of x with $\operatorname{supp}(x) = \{v \in V(\Gamma) \mid x(v) \neq 0\}$. In the following, we consider $u \in V(\Gamma)$ also as a function in $\ell^2(V(\Gamma))$ by setting $u(v) = 1$ if $u = v$, and $u(v) = 0$ if $u \neq v$. Furthermore, we extend this to finite subsets $U \subseteq V(\Gamma)$ by setting $U = \sum_{u \in U} u$. The norm of some linear operator $B : \ell^2(V(\Gamma)) \to \ell^2(V(\Gamma))$ is defined by $\|B\| = \sup\{\|Bx\|_2 \mid \|x\|_2 = 1\}$. Note that $\|B\| = \sup\{\|Bx\| \mid \|x\|_2 = 1, |\operatorname{supp}(x)| < \infty\}$. The *adjoint operator* of B is the uniquely defined operator B^* such that $\langle B^* x, y \rangle = \langle x, By \rangle$ for all $x, y \in \ell^2(V(\Gamma))$.

From now on, let $A : \ell^2(V(\Gamma)) \to \ell^2(V(\Gamma))$ denote the simple random walk operator of the d-regular graph Γ:

$$(Ax)(v) = \frac{1}{d} \sum_{\substack{e \in E(\Gamma) \\ \tau(e) = v}} x(\iota(e)).$$

For $n \in \mathbb{N}$ and $u, v \in V(\Gamma)$, we have

$$(A^n u)(v) = p^{(n)}(u, v).$$

Since Γ is undirected, A is symmetric, i. e., self-adjoint. The *spectral radius* $\rho(A)$ is defined as

$$\rho(A) = \sup \{\langle x,\, Ax\rangle \mid \|x\|_2 = 1\}.$$

We also write $\rho(\Gamma) = \rho(A)$. Because A is self-adjoint, it follows by standard facts:

Lemma 4.2. *If $P : \ell^2(V(\Gamma)) \to \ell^2(V(\Gamma))$ is some arbitrary linear operator and P^* denotes its adjoint operator, then $\|P^*P\| = \|P\|^2$. In particular, we have $\rho(A) = \|A\|$.*

Proof. See e. g. [HN01, Lem. 8.26]. □

The following theorem is well-known, see e. g. [CSGdlH99, Thm. 32, Thm. 51] and [Soa94, Thm. 4.27]. It goes back to Kesten, who gave a proof for Cayley graphs [Kes59a, Kes59b]. For arbitrary graphs of bounded degree, it appeared in [Ger88]. Condition (i) is due to Gromov (see [Gro93]).

Theorem 4.3 ([CSGdlH99, Soa94, Kes59a, Kes59b, Ger88, Gro93])**.** *Let Γ be a d-regular graph. Then the following statements are equivalent:*

(i) *There exists a map $f : V(\Gamma) \to V(\Gamma)$ such that $\sup_{v \in V(\Gamma)} d(f(v), v) < \infty$ and $|f^{-1}(v)| \geq 2$ for all $v \in V(\Gamma)$ (*Gromov condition*).*

(ii) *There exists some $k \in \mathbb{N}$ such that for every finite $U \subseteq V(\Gamma)$ we have*

$$\left|\mathcal{N}^k(U)\right| \geq 2\,|U|.$$

(iii) *Γ satisfies a strong isoperimetric inequality.*

(iv) *$\rho(\Gamma) < 1$.*

(v) *There is some $\sigma \in (0, 1)$ such that $p^{(n)}(u, v) \in o(\sigma^n)$ for all $u, v \in V(\Gamma)$.*

If Γ is connected, also the following condition is equivalent:

(vi) *There is some $\sigma \in (0, 1)$ and $u \in V(\Gamma)$ such that $p^{(n)}(u, u) \in o(\sigma^n)$.*

Theorem 4.3 holds, in fact, for arbitrary graphs with bounded degree, see [CSGdlH99] (this requires a slight change in the definition of the Hilbert space $\ell^2(V(\Gamma))$). For better readability and because it suffices for our purposes, we only state it for the d-regular case. In order to have a concise presentation we do not present a proof here (also, there is no new material contained in the proof). The reader who is interested in the details, can find a proof in Section B.1 in the appendix. Note that that proof can easily be transformed to work for all graphs of bounded degree.

Definition 4.4.

- A graph which meets one the conditions of Theorem 4.3 is called *non-amenable* (sometimes such a graph is also called an *infinite expander*, see [OW07]). Otherwise, it is called *amenable*.

- A finitely generated group is called *non-amenable* (resp. *amenable*) if it has a non-amenable (resp. amenable) Cayley graph.

Note that Definition 4.4 coincides with the common definition of amenability via invariant means, see e. g. [CSGdlH99] (originally due to Følner [Føl55]). Moreover, Corollary 4.11 below shows that for a group being amenable is independent of the generating set, and hence a well-defined property of the group.

Example 4.5. Non-abelian free groups are non-amenable. This can be verified e. g. using condition (i). The function f is defined by deleting the last letter of a freely reduced normal form.

Example 4.6. Groups of subexponential growth are amenable. (A finitely generated group G has subexponential growth if for every $\varepsilon > 0$ there is some N such that $|B(n)| = |\{g \in G \mid |g| \leq n\}| < (1+\varepsilon)^n$ for all $n \geq N$.) This is because the balls $B(n)$ already let the strong isoperimetric inequality fail. As a direct consequence, all virtually nilpotent and, in particular, all abelian groups are amenable.

The definition of amenability can be extended to directed graphs. However, one has to be careful as some of the conditions in Theorem 4.3 do not immediately make sense for directed graphs (although they could be adapted). Therefore, we consider condition (v) of Theorem 4.3 since the simple random walk (4.1) is also well-defined for directed graphs. As in the undirected case, the random walk starts in some vertex and then chooses every outgoing edge with the same probability. We say that the simple random walk (on a directed or undirected graph) has *exponentially decreasing return probability* if there is some $\sigma \in (0,1)$ such that $p^{(n)}(u,v) \in o(\sigma^n)$ for all $u, v \in V(\Gamma)$, where $p^{(n)}(u,v)$ is the probability that the simple random walk on Γ ends after n steps in v when starting in u.

Recall that a directed graph is d-regular if both the in-degree and out-degree of every vertex is d. For a directed graph $\Gamma = (V, E)$, we can construct an undirected graph $\Gamma' = (V, E \cup \overline{E})$, where $\overline{E}$ is a disjoint copy of E, and $\iota(\bar{e}) = \tau(e), \tau(\bar{e}) = \iota(e)$ for $\bar{e} \in \overline{E}$. If Γ is d-regular, then Γ' is $2d$-regular. We call Γ' the *undirected version* of Γ. Concerning simple random walks in directed graphs, we obtain Proposition 4.7, which is a special case of [Woe00, Thm. 10.6].

Proposition 4.7 ([Woe00])**.** *Let $\Gamma = (V, E)$ be some d-regular directed graph. If the undirected version $\Gamma' = (V, E \cup \overline{E})$ of Γ is non-amenable, then the simple random walk on Γ has exponentially decreasing return probability.*

Proof. We follow the proof as given in [Woe00]. Let B the simple random walk operator on Γ, i. e., as in the undirected case we have

$$(Bx)(v) = \frac{1}{d} \sum_{\substack{e \in E(\Gamma) \\ \tau(e) = v}} x(\iota(e)).$$

We set $P = \frac{1}{2}(\mathrm{id} + B)$, where id denotes the identity operator. That means P is the simple random walk operator of a graph which is obtained from Γ by putting d self-loops on every vertex. We set $Q = P^*P$, where P^* denotes the adjoint operator of P. Then Q is self-adjoint, and for all $(u, v) \in E(\Gamma')$, we have $\langle Qu, v\rangle > 0$. Moreover, for all $u, v \in V$, we have $\langle Qu, v\rangle \in \left(\frac{1}{2d}\right)^2 \cdot \mathbb{N}$, and

$$\sum_{u \in V} (Qv)(u) = 1 \qquad \text{for all } v \in V(\Gamma).$$

The last statement can be seen as follows: Let $x \in \mathbb{R}^V$ be some vector with $x \geq 0$ (componentwise). Then $\|Px\|_1 = \|x\|_1$ and $\|P^*x\|_1 = \|x\|_1$ because we assumed that every vertex of Γ has in-degree and out-degree d (with $\|x\|_1 = \sum_{v \in V(\Gamma)} |x(v)|$).

Hence, Q is the simple random walk operator of a (undirected) $\left(\frac{1}{2d}\right)^2$-regular graph Γ'' with $V(\Gamma'') = V$ and $E \cup \overline{E} = E(\Gamma') \subseteq E(\Gamma'')$. It is clear that if Γ' is non-amenable, then also Γ'' is non-amenable since adding additional edges does not make fail the strong isoperimetric inequality. Hence, by Lemma 4.2 and Theorem 4.3, it follows that $\|P\|^2 = \|Q\| < 1$. Since we have $(P^{2n}u)(v) = \sum_{i=0}^{2n} \binom{2n}{i} \frac{1}{2^{2n-i}} (B^i u)(v)$, we obtain

$$p^{(n)}(u, v) = (B^n u)(v) \leq 2n \cdot \left\|(P^{2n}u)(v)\right\| \leq 2n \cdot \left\|P^2\right\|^n \quad \text{with } \left\|P^2\right\| < 1. \qquad \square$$

Note that the converse of Proposition 4.7 is not true, in general (even not under the stronger assumption of Γ being strongly connected). For instance, consider the directed graph with vertex set $\mathbb{Z}$ and directed edges from n to $n - 1$ and $n + 2$. The simple random walk on this graph has exponentially decreasing return probability. However, the undirected version of this graph is amenable.

4.2. Quasi-isometries and Schreier Graphs

For a moment we leave the setting of d-regular graphs and consider, more generally, arbitrary metric spaces.

Definition 4.8. A *quasi-isometry* between two metric spaces (X, d_X) and (Y, d_Y) is a function $\varphi : X \to Y$ such that there is some constant $C > 0$ satisfying:

- $\frac{1}{C} \cdot d_X(x, y) - C \leq d_Y(\varphi(x), \varphi(y)) \leq C \cdot d_X(x, y) + C$ for all $x, y \in X$,
- for all $y \in Y$ there is some $x \in X$ such that $d_Y(y, \varphi(x)) \leq C$.

If there is a quasi-isometry between X and Y, then X and Y are called *quasi-isometric*. Two graphs Γ_1, Γ_2 are said to be quasi-isometric if the metric spaces $V(\Gamma_1)$ and $V(\Gamma_2)$ with the usual metric (assigning the length one to every edge) are quasi-isometric.

It is easy to see that being quasi-isometric is an equivalence relation. In particular, if $\varphi : X \to Y$ is a quasi-isometry, then there exists a quasi-isometry $\psi : Y \to X$. Furthermore, Cayley graphs of the same group with respect to different finite generating sets are quasi-isometric.

In [CSGdlH99, Prop. 38], it is stated that if two discrete metric spaces[1] (in particular, regular graphs) are quasi-isometric, then one of them is amenable if and only if the other is. In this case, amenability is defined via (i) or (ii) of Theorem 4.3, which are equivalent also in this more general case (see Section B.1 for the proof). Restated for a discrete metric space (X, d), these conditions are:

(i') There exists a map $f : X \to X$ such that $\sup_{x\in X} d(f(x), x) < \infty$ and $|f^{-1}(x)| \geq 2$ for all $x \in X$.

(ii') There exists some $k \in \mathbb{N}$ such that for every finite $U \subseteq X$ we have

$$\left|\mathcal{N}^k(U)\right| = |\{x \in X \mid \exists\, y \in U \,:\, d(x, y) \leq k\}| \geq 2\,|U|\,.$$

However, the statement in [CSGdlH99, Prop. 38] does not hold in this generality. In fact, we construct a locally finite graph which satisfies (i') and which is quasi-isometric to an amenable graph:

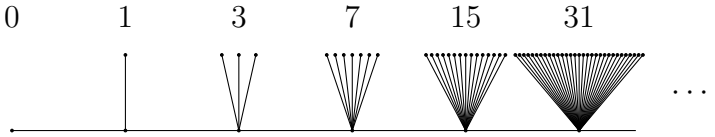


Figure 4.1.: Counterexample for [CSGdlH99, Prop. 38].

Example 4.9. Let Γ_1 and Γ_2 be simple graphs defined by

$$\begin{aligned}
V(\Gamma_1) &= \mathbb{N},\\
E(\Gamma_1) &= \left\{\{n, n+1\} \;\middle|\; n \in \mathbb{N}\right\},\\
V(\Gamma_2) &= \left\{(n, i) \;\middle|\; n \in \mathbb{N}, i \in \{0, \ldots, 2^n - 1\}\right\},\\
E(\Gamma_2) &= \left\{\{(n, 0), (n+1, 0)\} \;\middle|\; n \in \mathbb{N}\right\} \cup \left\{\{(n, 0), (n, i)\} \;\middle|\; n \in \mathbb{N}, 1 \leq i < 2^n\right\},
\end{aligned}$$

see Figure 4.1 for Γ_2. The map $\varphi : V(\Gamma_2) \to V(\Gamma_1)$ with $\varphi((n, i)) = n$ is obviously a quasi-isometry. Moreover, it is straightforward to see that Γ_1 is amenable since it does

[1] A discrete metric space is a metric space such that every ball of finite radius contains only finitely many points.

not satisfy the doubling condition (ii'). However, Γ_2 satisfies the Gromov condition (i') with the map $f : V(\Gamma_2) \to V(\Gamma_2)$ which is defined by

$$f((0,0)) = (0,0), \qquad f((n,i)) = (n-1, \lfloor i/2 \rfloor).$$

A correct statement of [CSGdlH99, Prop. 38] is as follows:

Proposition 4.10. *Let (X, d_X) and (Y, d_Y) be two discrete metric spaces. Additionally, assume that there exists a function $\xi : \mathbb{N} \to \mathbb{N}$ such that, for all $x \in X$, all $y \in Y$, and all $n \in \mathbb{N}$, we have $|\{z \mid d_X(x,z) \leq n\}| \leq \xi(n)$ and $|\{z \mid d_Y(y,z) \leq n\}| \leq \xi(n)$. Then X is amenable if and only if Y is amenable.*

Note that the condition $|\{z \mid d_X(x,z) \leq n\}| \leq \xi(n)$ is sometimes referred to as *bounded geometry*, see e. g. [Why98].

Proof. We are going to show that the doubling condition (ii') transfers from X to Y. Let $\varphi : Y \to X$ and $\psi : X \to Y$ be quasi-isometries. Because $|\{z \mid d_Y(y,z) \leq n\}| \leq \xi(n)$ and $|\{z \mid d_X(x,z) \leq n\}| \leq \xi(n)$ for all $x \in X$, $y \in Y$, and $n \in \mathbb{N}$, there is some constant κ such that $|\{z \in Y \mid \varphi(z) = x\}| \leq \kappa$ and $|\{z \in X \mid \psi(z) = y\}| \leq \kappa$ for all $x \in X$ and $y \in Y$.

Assume now that X satisfies the doubling condition (ii'). In particular, there is some constant k' such that, for every finite $U \subseteq X$, we have $\left|\mathcal{N}^{k'}(U)\right| \geq 2\kappa^2 |U|$. Moreover, we can find another constant ℓ such that, for every finite $U \subseteq Y$ the inclusion $\psi(\mathcal{N}^{k'}(\varphi(U))) \subseteq \mathcal{N}^{\ell}(U)$ holds. Hence, for every finite $U \subseteq Y$, we obtain

$$2|U| \leq 2\kappa |\varphi(U)| \leq \frac{\left|\mathcal{N}^{k'}(\varphi(U))\right|}{\kappa} \leq \left|\psi(\mathcal{N}^{k'}(\varphi(U)))\right| \leq \left|\mathcal{N}^{\ell}(U)\right|.$$

□

Now, as a consequence of Proposition 4.10, we obtain a special case of [CSGdlH99, Prop. 38], which is correct and which we also use later.

Corollary 4.11. *Let Γ_1 and Γ_2 be regular graphs (not necessarily the same degree) which are quasi-isometric. Then Γ_1 is amenable if and only if Γ_2 is amenable.*

In the following we want to investigate when groups are non-amenable "with respect to certain subgroups". In order to formalize this, we need a generalization of Cayley graphs, the so-called Schreier graphs (sometimes also Schreier coset graphs).

Definition 4.12. Let G be a finitely generated group, $H \leq G$ a subgroup and Σ a finite monoid generating set of G. The *directed Schreier graph* $\Gamma_{\text{dir}} = \Gamma_{\text{dir}}(G, H, \Sigma)$ of G with respect to H and Σ is defined as follows:

$$V(\Gamma) = {}_H\backslash G\,, \qquad E(\Gamma) = {}_H\backslash G \times \Sigma$$

with $\iota(Hg, a) = Hg$ and $\tau(Hg, a) = Hga$.
In the case that Σ is symmetric, the resulting graph is called *Schreier graph* $\Gamma = \Gamma(G, H, \Sigma)$ of ${}_H\backslash G$ with respect to Σ. In this case, we have $\overline{(Hg, a)} = (Hga, \bar{a})$.

We call a the *label* of the edge (Hg, a). Note that the subgroup $H \leq G$ acts on the Cayley graph $\Gamma(G, \Sigma)$, and the Schreier graph $\Gamma(G, H, \Sigma)$ can also be obtained as a quotient this action, i. e., $\Gamma(G, H, \Sigma) = {}_H\backslash\Gamma(G, \Sigma)$. In particular, if H is a normal subgroup, then $\Gamma(G, H, \Sigma) = \Gamma(G/_H, \Sigma)$.

Proposition 4.13 ([KMSS03, Prop 6.2])**.** *Let G be a finitely generated group and $H \leq G$ be a subgroup. Let Σ and Σ' be two finite symmetric generating sets of G. Then, $\Gamma(G, H, \Sigma)$ is amenable if and only if $\Gamma(G, H, \Sigma')$ is amenable.*
Moreover, if $\Gamma(G, H, \Sigma)$ is non-amenable and Σ'' is a finite monoid generating set of G, then the simple random walk on $\Gamma_{\text{dir}}(G, H, \Sigma'')$ has exponentially decreasing return probability.

Proof. The first statement is a direct consequence of Corollary 4.11 since the two Schreier graphs are quasi-isometric (as for Cayley graphs, the identity on the vertex sets is a quasi-isometry). The second statement follows from the first one and Proposition 4.7. □

Definition 4.14. Let Γ be a connected, d-regular graph. Let α_n be the number of paths without backtracking of length n starting at some fixed vertex v_0 and returning to v_0. The *cogrowth rate* $\alpha(\Gamma)$ is defined as:

$$\alpha(\Gamma) = \limsup_{n\to\infty} \sqrt[n]{\alpha_n}$$

Obviously, we have $\alpha(\Gamma) \leq d - 1$. Since Γ is assumed to be connected, $\alpha(\Gamma)$ is independent of the choice of the base point v_0. The next result, Theorem 4.15, was first proven independently by Cohen [Coh82] and Grigorchuk [Gri77] for Cayley graphs of finitely generated groups. The generalization to arbitrary d-regular graphs is due to Northshield [Nor92]. In [Bar99], Bartholdi developed some further generalizations dropping the regularity condition. Since these generalizations require some more definitions, we stick to the version by [Nor92].

Theorem 4.15 ([Bar99, Coh82, Gri77, Nor92])**.** *A connected d-regular graph is amenable if and only if its cogrowth rate is $d - 1$.*

4.3. Schreier Graphs of Amalgamated Products

Let G_1 and G_2 be groups intersecting in a common subgroup A. We consider the amalgamated product $G_1 *_A G_2$ – i. e., the special case of a graph of groups with two vertices and one edge connecting them (see Example 3.7). In this section, we need the convergent rewriting system $S_{\mathcal{G}}$ of Section 3.1 with two modifications: First, we change the "direction" of the rewriting system. Second, we can simplify the rules because there is no need for symbols for the edges of the underlying graph in the case of a single amalgamated product (i. e., we can work in the fundamental group with respect to some spanning tree).

The rewriting system we describe here, can also be found in [DDM10, Sec. 7.4]. We choose transversals $C_1 \subseteq G_2$ and $C_2 \subseteq G_2$ for cosets of A in G_1 and in G_2 with $1 \in C_1$

and $1 \in C_2$, such that there are unique decompositions $G_1 = AC_1$ and $G_2 = AC_2$. We let $\Delta = (G_1 \cup G_2) \setminus \{1\}$ and, as usual, we identify $1 \in G_1$ and $1 \in G_2$ with the empty word in Δ^*. As before, we use the convention to write $[ab]$ for the product ab whenever it is defined. The system $S \subseteq \Delta^2 \times (\{1\} \cup \Delta \cup \Delta^2)$ is defined by the following rules:

$$\begin{array}{rcll} gh & \longrightarrow & [gh] & \text{if } g,h \in \Delta \text{ such that } [gh] \text{ is defined,} \\ gh & \longrightarrow & [ga]c & \text{if } 1 \neq g \in G_1,\ a \in A,\ h \neq c \in C_2, \text{ and } h = [ac] \in G_2, \\ gh & \longrightarrow & [ga]c & \text{if } 1 \neq g \in G_2,\ a \in A,\ h \neq c \in C_1, \text{ and } h = [ac] \in G_1. \end{array}$$

The system defines the amalgamated product $G = G_1 *_A G_2$. Since S is essentially the system $S_{\mathcal{G}}$ of Section 3.1, we obtain the following lemma as a special case of Proposition 3.3 and Lemma 3.13 (iii) (see also [DDM10, Sec. 7.4]):

Lemma 4.16. *Each element of $G = G_1 *_A G_2$ is represented by a unique S-reduced word. Moreover, if $G_1 v =_G G_1 w$ and $v = gv'$ and $w = hw'$ are S-reduced words with $g, h \in G_1$ (possibly 1) and $v', w' \in \Delta^*$, then $v' = w'$.*

Let Σ be a finite symmetric generating set of G. In the following, we denote the index of a subgroup H in G with $[G : H]$.

Theorem 4.17. *Let $G = G_1 *_A G_2$ with $[G_1 : A], [G_2 : A] \geq 2$. Then the Schreier graph $\Gamma = \Gamma(G, G_1, \Sigma)$ is non-amenable if and only if $[G_1 : A] \geq 3$ or $[G_2 : A] \geq 3$.*

Proof. Let $[G_1 : A] = [G_2 : A] = 2$. Then A is normal in both G_1 and G_2, and hence in G. Therefore, the Schreier graph $\Gamma = \Gamma(G, G_1, \Sigma)$ is isomorphic to the Schreier graph $\Gamma(\mathbb{Z}/2\mathbb{Z} * \mathbb{Z}/2\mathbb{Z}, \mathbb{Z}/2\mathbb{Z}, \Sigma)$ and thus amenable. Now, let $[G_1 : A] \geq 3$. We show condition (i) in Theorem 4.3. In order to do so, we define a function $f : {}_{G_1}\backslash G \to {}_{G_1}\backslash G$ as follows: We fix some $C \subseteq C_1$ with $1 \notin C$ and $|C| = 2$ and some $1 \neq \tilde{c} \in C_2$. For $w \in \Delta^*$ and $c \in C$ such that $wc\tilde{c}$ is S-reduced, we set $f(G_1 wc\tilde{c}) = G_1 w$. Likewise, for $w \in \Delta^*$ and $c \in C$ such that $w\tilde{c}c$ is S-reduced, we set $f(G_1 w\tilde{c}c) = G_1 w$. Otherwise, we set $f(G_1 w) = G_1 w$. Because of Lemma 4.16, f is well-defined. Moreover, $\sup d(f(G_1 w), G_1 w) < \infty$ because C is finite (here, the distance d is defined w. r. t. Σ). Since $|C| = 2$ and for every word w either $wc\tilde{c}$ or $w\tilde{c}c$ is S-reduced, we have $|f^{-1}(G_1 w)| \geq 2$ for all $w \in G$. □

4.4. Schreier Graphs of HNN Extensions

Let $G = \langle H, t \mid tat^{-1} = \varphi(a) \text{ for } a \in A \rangle$ be an HNN extension of H with an isomorphism $\varphi : A \to B$ of subgroups $A, B \leq H$. Furthermore, let $\Sigma \subseteq H \cup \{t, \bar{t}\}$ be a finite symmetric generating set of G. We can prove an analog result as for amalgamated products:

Theorem 4.18. *Let $G = \langle H, t \mid tat^{-1} = \varphi(a), a \in A \rangle$. Then the Schreier graph $\Gamma = \Gamma(G, H, \Sigma)$ is non-amenable if and only if both $[H : A] \geq 2$ and $[H : \varphi(A)] \geq 2$.*

The proof works the same way as for amalgamated products. For completeness, we present it here. In Section B.2 in the appendix, we present another proof of Theorem 4.18, which uses random walks in the Schreier graph. That proof is self-contained and might

give an additional insight in generic sets. However, it is rather lengthy and complicated; therefore, we exclude it from the main part of this thesis.

In order to prove Theorem 4.18, we modify the convergent rewriting system $S_{\mathcal{G}}$ of Section 3.1 again. We choose transversals for cosets of A and $B = \varphi(A)$ in H; that is $C_A \subseteq H$ and $C_B \subseteq H$ with $1 \in C_A \cap C_B$ such that there are unique decompositions $H = AC_A$ and $H = BC_B$. We let $\Delta = (H \setminus \{1\}) \cup \{t, \bar{t}\}$ and we identify 1 with the empty word in Δ^*. The system $S \subseteq \Delta^2 \times (\{1\} \cup \Delta \cup \Delta^2)$ is defined by the following rules:

$$\begin{array}{rcll} t\bar{t} & \longrightarrow & 1 & \\ \bar{t}t & \longrightarrow & 1 & \\ gh & \longrightarrow & [gh] & \text{for } g, h \in H, \\ t[ac] & \longrightarrow & \varphi(a)tc & \text{if } 1 \neq a \in A,\ c \in C_A \\ \bar{t}[bc] & \longrightarrow & \varphi^{-1}(b)\bar{t}c & \text{if } 1 \neq a \in A,\ c \in C_B \end{array}$$

The system S defines the HNN extension $G = \langle H, t \mid tat^{-1} = \varphi(a), a \in A \rangle$ as it is simply the "opposite" system of $S_{\mathcal{G}}$. Since the system $S_{\mathcal{G}}$ of Section 3.1 is convergent, also S is confluent and terminating (see also [DDM10, Sec. 7.3]), and we have

Lemma 4.19. *Each $g \in G = \langle H, t \mid tat^{-1} = \varphi(a), a \in A \rangle$ is represented by a unique S-reduced word. Moreover, if $Hv =_G Hw$ and $v = gv'$ and $w = hw'$ are S-reduced words with $g, h \in H$ (possibly 1) and $v', w' \in \Delta^*$, then $v' = w'$.*

Proof of Theorem 4.18. First, consider the case $A = H$. We construct a sequence of finite sets $(U_n)_{n\in\mathbb{N}}$ such that $\lim_{n\to\infty} |\beta U_n| / |U_n| = 0$. This means the strong isoperimetric inequality does not hold. We set $U_n = \left\{ Ht^k \;\middle|\; 0 \leq k \leq n \right\} \subseteq V(\Gamma(G, H, \Sigma))$. Since $Ht^k a = Ht^k$ for all $a \in \Sigma \setminus \{t, \bar{t}\}$ and $k \geq 0$, the only edges leaving U_n are labeled with t or $\bar{t}$. But there are only two such edges leaving U_n. Hence, $|\beta U_n| \leq 4$ and $\lim_{n\to\infty} |\beta U_n| / |U_n| = 0$.

Now, let $[H : A] \geq 2$ and $[H : B] \geq 2$ for $B = \varphi(A)$. Again, we show condition (i) in Theorem 4.3. In order to do so, we define a function $f : {}_H\backslash G \to {}_H\backslash G$ as follows: We fix $1 \neq c_A \in C_A$ and $1 \neq c_B \in C_B$. Let $w \in \Delta^*$ be S-reduced. If w ends in t or $\bar{t}$, let $c \in \{c_A, c_B\}$ be such that wc is S-reduced. Then also wct and $wc\bar{t}$ are S-reduced and we set $f(Hwct) = f(Hwc\bar{t}) = Hw$. If w ends with some other letter or $w = 1$, then wt and $w\bar{t}$ are S-reduced and we set $f(Hwtc_A) = f(Hw\bar{t}c_B) = Hw$. Furthermore, we set $f(Hw') = Hw'$ for words w' which are not of the above form. Because of Lemma 4.19, f is well-defined and obviously $\sup d(f(Hw), Hw) < \infty$ (where d is defined w. r. t. Σ). Moreover, we have $|f^{-1}(Hw)| \geq 2$ for all $w \in G$. □

4.5. Amenability and Generic Sets

In this section, we want to show how to apply the results of the preceding sections of this chapter to strongly generic sets in HNN extensions and amalgamated products. Let G be a group with a finite monoid generating set Σ. As before, let $p^{(n)}(u, v)$ denote the probability that the simple random walk in the Cayley graph $\Gamma(G, \Sigma)$ ends in v after n

steps when starting in u. The following formula is just a reformulation of the definition of $p^{(n)}(u,v)$:

$$\frac{|\{w \in \Sigma^n \mid w =_G 1\}|}{|\Sigma^n|} = p^{(n)}(1,1).$$

Hence, in order to show that the set $\{w \in \Sigma^* \mid w \neq_G 1\}$ is (strongly) generic, we can look at the probability that the simple random walk returns to the origin. In the following chapters, we are interested in words which cannot be conjugated into the base group of some HNN extension. Therefore, we are rather interested in random walks in the Schreier graphs than in the Cayley graphs of these groups.

Let $G = \langle H, t \mid tat^{-1} = \varphi(a), a \in A \rangle$ be some HNN extension, Σ a finite monoid generating set of G, and $\eta : \Sigma^* \to G$ a presentation. We can rewrite any word $w \in \Sigma^*$ as a word $w' =_G w$ over some finite alphabet Σ' with $t, \bar{t} \in \Sigma'$ and $\Sigma' \setminus \{t, \bar{t}\} \subseteq H$. Now, if $|w'|_t \neq |w'|_{\bar{t}}$, then we know that w is not conjugate to any group element in H. Therefore, the probability that w is conjugate to some group element in H is tending to zero with increasing length of w. Hence, we have the following (trivial) result – compare to [KMSS03, Prop. 8.2].

Remark 4.20. The set $\Sigma^* \setminus \eta^{-1}\left(\bigcup_{g\in G} gHg^{-1}\right)$ is generic in Σ^*.

An analogous statement holds for amalgamated products $G = G_1 *_A G_2$ with $[G_1 : A], [G_2 : A] \geq 2$. However, to make some statements about strongly generic sets, we need some stronger assumptions and the results of the preceding sections. For the following result, recall that a word w over some symmetric alphabet Σ is called *cyclically freely reduced* if there is no factor $a\bar{a}$ in ww for any $a \in \Sigma$.

Theorem 4.21.

(i) *Let $G = \langle H, t \mid tat^{-1} = \varphi(a)$ for $a \in A \rangle$ be an HNN extension with $[H : A], [H : \varphi(A)] \geq 2$, and let $\eta : \Sigma^* \to G$ be a finite monoid presentation of G.*

  a) *The set $\Sigma^* \setminus \eta^{-1}\left(\bigcup_{g\in G} gHg^{-1}\right)$ is strongly generic in Σ^*.*

  b) *If Σ is symmetric and Ξ is the set of cyclically freely reduced words in Σ^*, then also $\Xi \setminus \eta^{-1}\left(\bigcup_{g\in G} gHg^{-1}\right)$ is strongly generic in Ξ.*

(ii) *Let $G = G_1 *_A G_2$ be an amalgamated product with $[G_1 : A] \geq 3$ and $[G_2 : A] \geq 2$, and let $\eta : \Sigma^* \to G$ be a finite monoid presentation of G.*

  a) *The set $\Sigma^* \setminus \eta^{-1}\left(\bigcup_{g\in G} gG_ig^{-1}\right)$ is strongly generic in Σ^* for $i = 1, 2$.*

  b) *If Σ is symmetric and Ξ is the set of cyclically freely reduced words in Σ^*, then also $\Xi \setminus \eta^{-1}\left(\bigcup_{g\in G} gG_ig^{-1}\right)$ is strongly generic in Ξ for $i = 1, 2$.*

Proof. We start with the proof of the statements (i) a) and (ii) a). The statements (i) b) and (ii) b) follow analogously using Theorem 4.15 – we give some details at the end.

Let Σ' be some finite symmetric generating set of G such that $t, \bar{t} \in \Sigma'$ and $\Sigma' \setminus \{t, \bar{t}\} \subseteq H$ (resp. $\Sigma' \subseteq G_1 \cup G_2$ for amalgamated products). For every $a \in \Sigma$, we fix some

$u_a \in \Sigma'^*$ with $a =_G u_a$. Now, if $w = w_1 \cdots w_n \in \Sigma^*$ with $w_i \in \Sigma$, then we can write $w =_G u_{w_1} \cdots u_{w_n} = u = u_1 \cdots u_m$ where $u \in \Sigma'^*$ is a word of length $m \in \Theta(n)$. By Theorem 3.19 and Lemma 3.18, we know that if w is conjugate to some element in H, then there is a cyclic permutation $u' = u_{i+1} \cdots u_m u_1 \cdots u_i$ of u with $u' \in_G H$.

For every $a \in \Sigma$ and every suffix $z \in \Sigma'^*$ of u_a, we choose some shortest $v,\tilde{v} \in \Sigma^*$ with $v =_G z$ and $\tilde{v} =_G z^{-1}$. Let c denote the maximal length of all these v, $\tilde{v}$. Now, there is some cyclic permutation $w_{j+1} \cdots w_n w_1 \cdots w_j$ of w and words $v, \tilde{v} \in \Sigma^*$ with $\tilde{v} =_G v^{-1}$ and $|v|$, $|\tilde{v}| \leq c$ such that

$$w' = v\, w_{j+1} \cdots w_n\, w_1 \cdots w_j\, \tilde{v} =_G u_{i+1} \cdots u_m\, u_1 \cdots u_i \in_G H. \tag{4.2}$$

Note that we have $n \leq |w'| \leq n + 2c$.

By Theorem 4.18 (resp. Theorem 4.17 for amalgamated products), we know that the Schreier graph $\Gamma(G, H, \Sigma')$ is non-amenable. By Proposition 4.13, this implies that there is some $\sigma \in (0, 1)$ such that $p^{(n)}(u, v) \in o(\sigma^n)$, where $p^{(n)}(u, v)$ is the probability distribution of the simple random walk on the directed Schreier graph $\Gamma_{\mathrm{dir}}(G, H, \Sigma)$. In particular, we have

$$\frac{|\{w \in \Sigma^n \mid w \in_G H\}|}{|\Sigma^n|} = p^{(n)}(H, H) \in o(\sigma^n).$$

Hence, for n large enough, there are at most $(|\Sigma| \cdot \sigma)^n$ words of length between n and $n + 2c$ which represent group elements in H. For each of these words, there are at most $(c + 1) \cdot n$ conjugates of the form (4.2) with length n (given some w', there are at most $c + 1$ possibilities for v – each of which also determines $\tilde{v}$). Thus, we obtain

$$\begin{aligned}
&\frac{|\{w \in \Sigma^n \mid w \sim_G h \text{ for some } h \in H\}|}{|\Sigma^n|} \\
&\qquad \leq (c + 1) \cdot n \cdot \frac{|\{w' \in \Sigma^* \mid w' \in_G H, n \leq |w'| \leq n + 2c\}|}{|\Sigma^n|} \\
&\qquad \leq (c + 1) \cdot n \cdot \sigma^n.
\end{aligned}$$

To prove the statement about freely reduced words, we proceed as follows: We apply free reductions to w' of (4.2) leading to a word w'' which is freely reduced and such that $w'' \sim_G w$ and $n - 2c \leq |w''| \leq n + 2c$. By Theorem 4.15 and Theorem 4.18 (resp. Theorem 4.17), we have

$$\frac{|\{w \in \Xi \mid w \in_G H, |w| = n\}|}{|\{w \in \Xi \mid |w| = n\}|} \in o\left(\left(\frac{d - 1 - \varepsilon}{d - 1}\right)^n\right).$$

for some $\varepsilon \in (0, 1)$. We set $\sigma = \frac{d-1-\varepsilon}{d-1}$. Now, the proof continues as in the other case: For n large enough, there are at most $((d - 1) \cdot \sigma)^n$ freely reduced words of length between $n - 2c$ and $n + 2c$ which represent group elements in H. Moreover, at most $|\Sigma|^{2c+2} \cdot (c + 1) \cdot n$ (only a rough upper bound) cyclically freely reduced words of length n can be freely reduced to each of these after a conjugation of the form (4.2). Thus, as before, we obtain the desired result. □

Chapter 5.

Free (Abelian) Vertex Groups

In this chapter, we give some undecidability and decidability results for fundamental groups of finite graphs of groups with free and free abelian vertex groups. For now, our main focus is on decidability and not on complexity. We start by presenting a construction for groups with undecidable conjugacy problem – similar as Miller's group [Mil71], but leading also to undecidability results for HNN extensions of free abelian groups. After that, we consider some special cases, where the conjugacy problem can be decided.

5.1. Some Undecidability Results

In 1971, Miller constructed an amalgamated product of two free groups with undecidable conjugacy problem [Mil71]. Later, in [Sti82], Stillwell gave a simple construction for an HNN extension of a free group with undecidable subgroup membership problem by starting with a universal Turing machine. It turns out that the same group has undecidable conjugacy problem. Hence, this gives another (presumably easier) proof for Miller's result that a fundamental group of a finite graph of groups with free vertex groups can have undecidable conjugacy problem [Mil71].

In this section, we present Stillwell's construction. This construction also gives rise to an HNN extension of a free abelian group of rank three with undecidable conjugacy problem. After that, we continue with that group and develop an HNN extension of a free abelian group with only two stable letters but undecidable conjugacy problem (Section 5.1.2). Finally, we construct an HNN extension of a free group of rank two with a single stable letter as well as an amalgamated product of two free groups of rank two with undecidable conjugacy problem (Section 5.1.3).

Note that in [Loc89], Lockhart constructed an HNN extension with cyclic associated subgroup which has undecidable conjugacy problem, although the base group has decidable conjugacy problem. However, the base group is not finitely presented.

5.1.1. Stillwell's Construction

We start with Stillwell's construction for an HNN extension with undecidable conjugacy problem. In order to do so, we need a formal definition of a Turing machine.

A *Turing machine* is given by a 7-tuple $\mathcal{M} = (Q, \Sigma, \Gamma, \rho, q_0, \square, F)$, where Q is the set of *states*, $\Sigma \subseteq \Gamma$ are the *input* resp. *tape alphabet*, $\rho \subseteq Q \times \Gamma \times Q \times \Gamma \times \{L, N, R\}$

the *transition relation*, $q_0 \in Q$ the *initial state*, $\square \in \Gamma$ the *blank symbol*, and $F \subseteq Q$ the set of *final states*. We make some additional assumptions:

(i) $\mathcal{M}$ is deterministic, i. e., ρ is a function $Q \times \Gamma \to Q \times \Gamma \times \{L, N, R\}$.

(ii) There is some number $M \in \mathbb{N}$ with $Q, \Gamma \subseteq \{0, \dots, M-1\}$ and $\square = 0$.

(iii) There is only one final state 0.

(iv) Before $\mathcal{M}$ goes into the final state, it deletes all symbols on the tape.

(v) Once in the final state, $\mathcal{M}$ never leaves the final state.

It is straightforward to transform an arbitrary deterministic Turing machine into one that meets these additional requirements.

A configuration of $\mathcal{M}$ is a word $u_m \cdots u_1 q\, v_0 v_1 \cdots v_n$ where $u = u_m \cdots u_1$ is the non-blank word on the tape left of the current position of the read-write-head, q is the current state, v_0 the symbol at the current position, and $v_1 \cdots v_n$ the non-blank word on the tape right of the current position. For two configurations $uqv, u'q'v'$, we write $uqv \vdash u'q'v'$ if there is a transition leading from uqv to $u'q'v'$. Furthermore, $\vdash^*$ denotes the reflexive and transitive closure of $\vdash$, i. e., $uqv \vdash^* u'q'v'$ if and only if there is a computation of $\mathcal{M}$ transforming uqv into $u'q'v'$.

We can encode a configuration $u_m \cdots u_1 q v_0 v_1 \cdots v_n$ of $\mathcal{M}$ by a pair $(\mu, \nu) \in \mathbb{N}^2$, where for $i > m$, we set $u_i = 0$, resp. $v_i = 0$ for $i > n$:

$$\mu = q + \sum_{i \geq 1} u_i M^i, \qquad \nu = \sum_{i \geq 0} v_i M^i. \tag{5.1}$$

Note that these sums are always finite since there are always only finitely many non-blank symbols on the tape. Moreover, by our assumptions, the only accepting configuration is $(0, 0)$. A transition $(q, \beta, q', \beta', \cdot) \in \rho$ can be applied to a configuration (μ, ν) if and only if $\mu \equiv q \mod M$ and $\nu \equiv \beta \mod M$.

Now, let $\mathcal{L}$ denote the set of transitions $(q, \beta, q', \beta', L) \in \rho$, $\mathcal{N}$ the set of transitions $(q, \beta, q', \beta', N) \in \rho$, and $\mathcal{R}$ the set of transitions $(q, \beta, q', \beta', R) \in \rho$. That means we split the transitions into three disjoint sets $\rho = \mathcal{L} \cup \mathcal{N} \cup \mathcal{R}$. The transitions have the following effect on the numerical encodings:

(i) For every $\ell \in \mathcal{L}$, there are unique $\alpha_\ell, \beta_\ell, \gamma_\ell, \delta_\ell \in \{0, \dots, M-1\}$ such that

$$(\mu M^2 + \zeta M + \alpha_\ell,\ \nu M + \beta_\ell) \vdash (\mu M + \gamma_\ell,\ \nu M^2 + \delta_\ell M + \zeta)$$

for all $\zeta \in \{0, \dots, M-1\}$.

(ii) For every $n \in \mathcal{N}$, there are unique $\alpha_n, \beta_n, \gamma_n, \delta_n \in \{0, \dots, M-1\}$ such that

$$(\mu M + \alpha_n, \nu M + \beta_n) \vdash (\mu M + \gamma_n, \nu M + \delta_n).$$

(iii) For every $r \in \mathcal{R}$, there are unique $\alpha_r, \beta_r \in \{0, \ldots, M-1\}$, $\gamma_r \in \{0, \ldots, M^2-1\}$ such that
$$(\mu M + \alpha_r, \nu M + \beta_r) \vdash (\mu M^2 + \gamma_r, \nu).$$

Now, let us turn to groups. We consider the free group $F = F_{\{a,b,c\}}$ and its quotient, the free abelian group $H = \mathbb{Z}^3$, with $a \mapsto (1,0,0)$, $b \mapsto (0,0,1)$, and $c \mapsto (0,1,0)$. In both groups, every configuration (μ, ν) of the Turing machine $\mathcal{M}$ can be uniquely encoded as the group element $a^\mu c b^\nu$, resp. $(\mu, 1, \nu)$. Moreover, we can describe transitions as isomorphisms of subgroups of F, resp. H. For every $\ell \in \mathcal{L}$, $\zeta \in \Gamma$, we define a homomorphism

$$\begin{aligned}\varphi_{(\ell,\zeta)} : \left\langle a^{M^2}, b^M, a^{\zeta M + \alpha_\ell} c b^{\beta_\ell} \right\rangle &\to \left\langle a^M, b^{M^2}, a^{\gamma_\ell} c b^{\delta_\ell M + \zeta} \right\rangle, \\ a^{M^2} &\mapsto a^M, \\ b^M &\mapsto b^{M^2}, \\ a^{\zeta M + \alpha_\ell} c b^{\beta_\ell} &\mapsto a^{\gamma_\ell} c b^{\delta_\ell M + \zeta}.\end{aligned}$$

In fact, these homomorphisms are isomorphisms between subgroups since both $\left\{a^{M^2}, b^M, a^{\zeta M + \alpha_\ell} c b^{\beta_\ell}\right\}$ and $\left\{a^M, b^{M^2}, a^{\gamma_\ell} c b^{\delta_\ell M + \zeta}\right\}$ are bases of free subgroups of rank three. Likewise for $n \in \mathcal{N}$ and $r \in \mathcal{R}$ we define isomorphisms

$$\begin{aligned}\varphi_n : \left\langle a^M, b^M, a^{\alpha_n} c b^{\beta_n} \right\rangle &\to \left\langle a^M, b^M, a^{\gamma_n} c b^{\delta_n} \right\rangle, \\ a^M &\mapsto a^M, \\ b^M &\mapsto b^M, \\ a^{\alpha_n} c b^{\beta_n} &\mapsto a^{\gamma_n} c b^{\delta_n},\end{aligned}$$

and

$$\begin{aligned}\varphi_r : \left\langle a^M, b^M, a^{\alpha_r} c b^{\beta_r} \right\rangle &\to \left\langle a^{M^2}, b, a^{\gamma_r} c \right\rangle, \\ a^M &\mapsto a^{M^2}, \\ b^M &\mapsto b, \\ a^{\alpha_r} c b^{\beta_r} &\mapsto a^{\gamma_r} c.\end{aligned}$$

We denote the set of all these isomorphisms of subgroups of F with $\Phi_F = \{\varphi_d \mid d \in (\mathcal{L} \times \Gamma) \cup \mathcal{N} \cup \mathcal{R}\}$. Obviously, all these isomorphisms induce isomorphisms of subgroups of rank three of H. We denote the set of these induced isomorphisms by Φ_H. By the definition of Φ_F, it follows that $(\mu, \nu) \vdash (\mu', \nu')$ implies that there is some $\varphi \in \Phi_F$ with $\varphi(a^\mu c b^\nu) = a^{\mu'} c b^{\nu'}$. Recall that $\mathrm{dom}(\varphi)$ denotes the domain of φ.

Lemma 5.1 ([Sti82, Lem. 1])**.** *For every $(\mu, 1, \nu) \in H$ with $\mu, \nu \in \mathbb{Z}$, there is at most one $\varphi \in \Phi_H$ with $(\mu, 1, \nu) \in \mathrm{dom}(\varphi)$. If $\varphi((\mu, 1, \nu)) = (\mu', 1, \nu')$ for some $\varphi \in \Phi_H$, then $(\mu, \nu) \vdash (\mu', \nu')$. Moreover, if $\varphi^{-1}((\mu, 1, \nu)) = (0, 1, 0)$, then $\mu = \nu = 0$.*

Proof. Let $d \in (\mathcal{L} \times \Gamma) \cup \mathcal{N} \cup \mathcal{R}$. For the first statement, note that $(\mu, 1, \nu) \in \mathrm{dom}(\varphi_d)$ implies that $\mu \equiv \alpha_d \mod M$ and $\nu \equiv \beta_d \mod M$ (if $d = (\ell, \zeta)$, we set $\alpha_d = \alpha_\ell$, etc.). Since $\mathcal{M}$ is assumed to be deterministic, there is at most one transition satisfying this condition. As $\mu \equiv \alpha_d \mod M$ and $\nu \equiv \beta_d \mod M$, the transition d can be applied to (μ, ν), and $(\mu, \nu) \vdash (\mu', \nu')$ if $\varphi_d((\mu, 1, \nu)) = (\mu', 1, \nu')$.

The last statement is an immediate consequence of the first statement and the assumption that $\mathcal{M}$ never leaves the accepting configuration $(0, 0)$. □

Now, we introduce HNN extensions G and $\widetilde{G}$ of F resp. H:

$$G = \left\langle F, (t_\varphi)_{\varphi \in \Phi_F} \;\middle|\; t_\varphi a t_\varphi^{-1} = \varphi(a) \text{ for } \varphi \in \Phi_F, a \in \mathrm{dom}(\varphi) \right\rangle,$$
$$\widetilde{G} = \left\langle H, (t_\varphi)_{\varphi \in \Phi_H} \;\middle|\; t_\varphi a t_\varphi^{-1} = \varphi(a) \text{ for } \varphi \in \Phi_H, a \in \mathrm{dom}(\varphi) \right\rangle.$$

Note that there is a canonical epimorphism $G \to \widetilde{G}$.

Lemma 5.2. *Let $(\mu, 1, \nu) \sim_{\widetilde{G}} (0, 1, 0)$. Then there is some $w \in \{t_\varphi \mid \varphi \in \Phi_H\}^*$ such that $w \cdot (\mu, 1, \nu) \cdot w^{-1} =_{\widetilde{G}} (0, 1, 0)$. In particular, w does not use any letter t_φ^{-1}.*

Proof. Let $(\mu, 1, \nu) \sim_{\widetilde{G}} (0, 1, 0)$ and $(\mu, 1, \nu) \neq_{\widetilde{G}} (0, 1, 0)$. We are in case (i) or (ii) of the Conjugacy Criterion, Theorem 3.19. Since H is abelian, a conjugation by some element of H has no effect. Hence, there is some $w \in \left\{t_\varphi^{\pm 1} \;\middle|\; \varphi \in \Phi_H\right\}^*$ such that $w \cdot (\mu, 1, \nu) \cdot w^{-1} =_{\widetilde{G}} (0, 1, 0)$ in $\widetilde{G}$. Choose a shortest such w and assume that it contained some letter t_φ^{-1}. Then we can write $w = w' t_\varphi^{-1} w''$ for some $w' \in \{t_\varphi \mid \varphi \in \Phi_H\}^*$ and $w'' \in \left\{t_\varphi^{\pm 1} \;\middle|\; \varphi \in \Phi_H\right\}^*$. Because of the last statement in Lemma 5.1, we know that $w' \neq 1$. Since $t_\varphi^{-1} w'' \cdot (\mu, 1, \nu) \cdot w''^{-1} t_\varphi \in \mathrm{dom}(\varphi)$ and $w' \cdot (t_\varphi^{-1} w'' \cdot (\mu, 1, \nu) \cdot w''^{-1} t_\varphi) \cdot w'^{-1} = (0, 1, 0)$ in $\widetilde{G}$ and the uniqueness statement in Lemma 5.1, it follows that w' ends in t_φ. However, this is a contradiction to the assumption that w was shortest possible. Hence, w does not contain any letter t_φ^{-1}. □

Proposition 5.3. *The following statements are equivalent:*

(i) $(\mu, \nu) \vdash^* (0, 0)$.

(ii) $a^\mu c b^\nu \sim_G c$.

(iii) $(\mu, 1, \nu) \sim_{\widetilde{G}} (0, 1, 0)$.

Proof. Let $(\mu, \nu) \vdash^* (0, 0)$, i. e., there is a sequence of transitions $d_1, \ldots, d_k$ transforming (μ, ν) into $(0, 0)$. Hence, the sequence of isomorphisms $\varphi_{d_1}, \ldots, \varphi_{d_k}$ transforms $a^\mu c b^\nu$ into c, and thus by the definition of G we have $a^\mu c b^\nu \sim_G c$.

Since $\widetilde{G}$ is a quotient of G, the implication (ii) ⇒ (iii) is trivial. If $(\mu, 1, \nu) \sim_{\widetilde{G}} (0, 1, 0)$, then, by Lemma 5.2, there is some sequence of isomorphisms $\varphi_1, \ldots, \varphi_k$ with $\varphi_i \in \Phi_H$ which transforms $(\mu, 1, \nu)$ into $(0, 1, 0)$. Hence, by Lemma 5.1, $(\mu, \nu) \vdash^* (0, 0)$. □

5.1.2. Undecidability with Free Abelian Vertex Groups

In [BMV10], Bogopolski, Martino and Ventura construct an HNN extension of a free abelian group of rank four with undecidable conjugacy problem (in fact, they construct a $\mathbb{Z}^4$-by-free group). In [Bee11b], the existence of an HNN extension of a free abelian group of rank three is listed as open problem (see [Bee11a] for the English translation).

Theorem 5.4. *There is an HNN extension (with more than one stable letter) of a free abelian group of rank three with undecidable conjugacy problem.*

Proof. Just take any Turing machine with undecidable halting problem and apply Proposition 5.3. □

In Theorem 5.4, we use many stable letters in order to achieve the undecidability result. The next result is a construction showing that actually two stable letters are sufficient (however, at the cost of increasing the rank of the base group). Proposition 5.18 below shows that an HNN extension of a free abelian group with one stable letter has decidable conjugacy problem. Hence, both Proposition 5.18 and Theorem 5.5 are somehow optimal.

Theorem 5.5. *There is a free abelian group H' with subgroups $A, B, C, D \leq H'$ and two isomorphisms $\varphi : A \to B$ and $\psi : C \to D$ such that the HNN extension $\langle H', s, t \mid sas^{-1} = \varphi(a), tct^{-1} = \psi(t)$ for $a \in A, c \in C \rangle$ has undecidable conjugacy problem.*

Note that Theorem 5.5 is similar to the result of [BMV10] which states that there is some semidirect product $\mathbb{Z}^4 \rtimes F_n$ with undecidable conjugacy problem. This means there is an HNN extension with several stable letters which has undecidable conjugacy problem. In fact, if the result in [BMV10] were true for $n = 2$, then it would imply Theorem 5.5.

Proof. Recall the HNN extension with undecidable conjugacy problem

$$\widetilde{G} = \left\langle H, (t_\varphi)_{\varphi \in \Phi_H} \;\middle|\; t_\varphi a t_\varphi^{-1} = \varphi(a) \text{ for } \varphi \in \Phi_H, a \in \mathrm{dom}(\varphi) \right\rangle.$$

where $H = \mathbb{Z}^3$ is free abelian. Let $\Phi_H = \{\varphi_1, \ldots, \varphi_k\}$, i. e., we identify Φ_H of Section 5.1.1 with the set of natural numbers $\{1, \ldots, k\}$ for some $k \in \mathbb{N}$. We set $H' = H^k$ with componentwise multiplication. Now, we can define an automorphism on H' by $\psi((h_1, \ldots, h_k)) = (h_k, h_1, \ldots, h_{k-1})$, i. e., the components are shifted cyclically. Moreover, we define $A = \{(h_1, \ldots, h_k) \in H' \mid h_i \in \mathrm{dom}(\varphi_i) \text{ for all } i\}$ and $\varphi((h_1, \ldots, h_k)) = (\varphi_1(h_1), \ldots, \varphi_k(h_k))$ for $(h_1, \ldots, h_k) \in A$. Finally, we set

$$G' = \left\langle H, s, t \;\middle|\; sas^{-1} = \varphi(a),\ tht^{-1} = \psi(h) \text{ for } a \in A, h \in H' \right\rangle.$$

Now it is straightforward to see that $(\mu, 1, \nu) \sim_{\widetilde{G}} (0, 1, 0)$ if and only if

$$((\mu, 1, \nu), (0, 0, 0), \ldots, (0, 0, 0)) \sim_{G'} ((0, 1, 0), (0, 0, 0), \ldots, (0, 0, 0)).$$ □

5.1.3. Undecidability with Free Vertex Groups

Analogously to Theorem 5.4 we obtain the following theorem from the results of Section 5.1.1 for the group G.

Theorem 5.6 ([Mil71])**.** *There is an HNN extension (with more than one stable letter) of a f. g. free group with f. g. edge groups which has undecidable conjugacy problem.*

Like in Theorem 5.4 we use more than one stable letter in order to achieve the undecidability result in Theorem 5.6. In the following, we want to embed the group G of Section 5.1.1 into a single HNN extension of a free group and also into a single amalgamated product of two free groups. First, we apply the same construction Miller used to embed his HNN group with undecidable conjugacy problem into an amalgamated product of two free groups, [Mil71]. Then, in a second step we embed the resulting groups in an amalgamated product resp. HNN extension of free groups of rank two.

Recall that the group G is the HNN extension

$$G = \left\langle F, (t_\varphi)_{\varphi\in\Phi_F} \;\middle|\; t_\varphi a t_\varphi^{-1} = \varphi(a) \text{ for } \varphi \in \Phi_F, a \in \operatorname{dom}(\varphi) \right\rangle$$

where F is a free group. Let $\Phi_F = \{\varphi_1, \ldots, \varphi_k\}$, $A_i = \operatorname{dom}(\varphi_i)$, and $B_i = \varphi_i(A_i)$ for $1 \leq i \leq k$. We construct an amalgamated product and an HNN extension of the free group $F' = \langle F, d_1, \ldots, d_k \mid \rangle = F * F_{\{d_1,\ldots,d_k\}}$ where $d_1, \ldots, d_k$ are new letters. Let $A, B \subseteq F'$ be the subgroups

$$A = \left\langle F, d_1 A_1 d_1^{-1}, d_2 A_2 d_2^{-1}, \ldots, d_k A_k d_k^{-1} \right\rangle,$$
$$B = \left\langle F, d_1 B_1 d_1^{-1}, d_2 B_2 d_2^{-1}, \ldots, d_k B_k d_k^{-1} \right\rangle,$$

and let φ be the isomorphism $A \to B$ being the identity on F and $\varphi(d_i a d_i^{-1}) = d_i \varphi_i(a) d_i^{-1} \in d_i B_i d_i^{-1}$ for $a \in A_i$ (since φ induces an isomorphisms $d_i A_i d_i^{-1} \to d_i B_i d_i^{-1}$ for all i, it is, in fact, an isomorphism). Now, we can build an amalgamated product and an HNN extension of F':

$$G_{\text{am}} = F' *_{A=\varphi(A)} F', \tag{5.2}$$
$$G_{\text{HNN}} = \left\langle F', t \;\middle|\; t a t^{-1} = \varphi(a) \text{ for } a \in A \right\rangle. \tag{5.3}$$

Following [Mil71], we apply Tietze transformations in order to understand the structure of G_{am} and G_{HNN} better. Here e_i denotes the letter d_i of the right factor of the amalgamated product, whereas d_i is the d_i in the left factor:

$$\begin{aligned}
G_{\text{am}} &= \left\langle F, d_1, \ldots, d_k, e_1, \ldots, e_k \;\middle|\; d_i a d_i^{-1} = e_i \varphi_i(a) e_i^{-1} \text{ for } i \in \{1, \ldots, k\}, a \in A_i \right\rangle \\
&= \langle F, d_1, \ldots, d_k, e_1, \ldots, e_k, t_1, \ldots, t_k \mid \\
&\qquad\qquad d_i a d_i^{-1} = e_i \varphi_i(a) e_i^{-1}, t_i = e_i^{-1} d_i \text{ for } i \in \{1, \ldots, k\}, a \in A_i \rangle \\
&= \langle F, d_1, \ldots, d_k, e_1, \ldots, e_k, t_1, \ldots, t_k \mid \\
&\qquad\qquad t_i a t_i^{-1} = \varphi_i(a), t_i = e_i^{-1} d_i \text{ for } i \in \{1, \ldots, k\}, a \in A_i \rangle \\
&= \left\langle F, d_1, \ldots, d_k, t_1, \ldots, t_k \;\middle|\; t_i a t_i^{-1} = \varphi_i(a) \text{ for } i \in \{1, \ldots, k\}, a \in A_i \right\rangle \\
&= G * F_{\{d_1,\ldots,d_k\}}.
\end{aligned}$$

Likewise, we obtain

$$\begin{aligned}
G_{\text{HNN}} &= \left\langle F, d_1, \dots, d_k, t \;\middle|\; td_iad_i^{-1}t^{-1} = d_i\varphi_i(a)d_i^{-1} \text{ for } i \in \{1,\dots,k\}, a \in A_i \right\rangle \\
&= \langle F, d_1, \dots, d_k, t, t_1, \dots, t_k \mid \\
&\qquad\qquad td_iad_i^{-1}t^{-1} = d_i\varphi_i(a)d_i^{-1}, t_i = d_i^{-1}td_i \text{ for } i \in \{1,\dots,k\}, a \in A_i \rangle \\
&= \langle F, d_1, \dots, d_k, t, t_1, \dots, t_k \mid \\
&\qquad\qquad t_iat_i^{-1} = \varphi_i(a), t_i = d_i^{-1}td_i \text{ for } i \in \{1,\dots,k\}, a \in A_i \rangle \\
&= \left\langle F, t, t_1, \dots, t_k \;\middle|\; t_iat_i^{-1} = \varphi_i(a) \text{ for } i \in \{1,\dots,k\}, a \in A_i \right\rangle \\
&= G * \langle t \rangle .
\end{aligned}$$

Hence, G_{am} and G_{HNN} are free products of G with some free group. It is easy to see (and a direct consequence of the Conjugacy Criterion, Theorem 3.19), that $g \sim_G h$ if and only if $g \sim_{G_{\text{am}}} h$ resp. $g \sim_{G_{\text{HNN}}} h$ for $g, h \in G$.

In order to prove the next theorem, we need the following lemma. Recall that a subgroup $F \leq G$ is called *malnormal* if and only if for all $g \in G \setminus F$ the subgroup $F \cap gFg^{-1}$ is the trivial group.

Lemma 5.7. *Let F' be a free group, $A, B \leq F'$, and $\varphi : A \to B$ an isomorphism. Let $\zeta : F' \to F$ be an embedding of F' into another free group F such that $\zeta(F')$ is malnormal in F.*

(*a*) *If $G = F' *_{A=B} F'$ is an amalgamated product, then ζ extends to a canonical embedding of G into*

$$H = F *_{\zeta(A)=\zeta(B)} F$$

and for $g_1, g_2 \in G$ we have $g_1 \sim_G g_2$ if and only if $\zeta(g_1) \sim_H \zeta(g_2)$.

(*b*) *If $G = \langle F', t \mid tat^{-1} = \varphi(a) \text{ for } a \in A \rangle$ is an HNN extension, then ζ extends to a canonical embedding of G into the HNN extension*

$$H = \left\langle F, t \;\middle|\; tat^{-1} = \varphi(a) \text{ for } a \in \zeta(A) \right\rangle$$

and for $g_1, g_2 \in G$ we have $g_1 \sim_G g_2$ if and only if $\zeta(g_1) \sim_H \zeta(g_2)$.

Proof. In both cases, injectivity of the induced homomorphism $G \to H$ is by Proposition 3.11. We also denote this induced homomorphism by ζ. We consider case (a), only. Let $g_1, g_2 \in G$ with $\zeta(g_1) \sim_H \zeta(g_2)$. W. l. o. g. we may assume that g_1, g_2 are cyclically reduced. This implies immediately that $\zeta(g_1)$ and $\zeta(g_2)$ are cyclically reduced. We distinguish the three cases of Theorem 3.19:

First, let $\zeta(g_1), \zeta(g_2) \in F \cap \zeta(G)$ (either the left or right factor F of the amalgamated product) be conjugate to some element of $\zeta(A_\ell) =_G \zeta(B_r)$ where A_ℓ (resp. B_r) denotes the subgroup A (resp. B) of the left (resp. right) factor F'. By the Conjugacy Criterion, Theorem 3.19, there is a sequence $h_1, \dots, h_k \in \zeta(A_\ell) \cup \zeta(B_r)$ such that $g_1 \sim_F h_1$, $h_k \sim_F g_2$ and for all i either $h_i \sim_F h_{i-1}$ or $h_i = \varphi^{\pm 1}(h_{i-1})$. Because $\zeta(F')$ is malnormal in F, we have $\zeta(g) \sim_F \zeta(h)$ if and only if $g \sim_{F'} h$ for all $g, h \in F'$. Hence, $g_1 \sim_G g_2$.

In case (ii) of Theorem 3.19, we have $\zeta(g_1) \sim_F \zeta(g_2)$, what, by malnormality of $\zeta(F')$, implies that $\zeta(g_1) \sim_{F'} \zeta(g_2)$.

Finally, $\zeta(g_1), \zeta(g_2) \notin F$. Then, by the Conjugacy Criterion, Theorem 3.19, $\zeta(g_1)$ can be obtained from $\zeta(g_2)$ by a cyclic permutation followed by a conjugation by some element in $\zeta(A)$. This, in turn, means that g_1 and g_2 are conjugate in G.

The proof for the HNN case (b) follows analogously. □

To the best of our knowledge, the next result has not been stated before, although, by Lemma 5.7, it follows rather easily from [Mil71].

Theorem 5.8. *There are finitely generated subgroups $A, B \leq F_2$ and an isomorphism $\varphi : A \to B$ such that the amalgamated product $F_2 *_{A=B} F_2$ and the HNN extension $\langle F_2, t \mid tat^{-1} = \varphi(a)$ for $a \in A \rangle$ have undecidable conjugacy problem.*

Compare Theorem 5.8 to Corollary 6.22 and Corollary 7.2 of [BMV10]. There the situation with various stable letters and $A = B = F_m$ is considered: An HNN extension $\left\langle F_m, t_1, \ldots, t_k \;\middle|\; t_i a t_i^{-1} = \varphi_i(a), \text{ for } i = 1, \ldots, k,\ a \in F_m \right\rangle$ has decidable conjugacy problem for $m = 2$, whereas it can have undecidable conjugacy problem for $m \geq 3$. By Theorem 5.8, this change from decidable to undecidable happens for arbitrary inclusions of A, B in F_m between $m = 1$ and $m = 2$ – also for only one stable letter – what is not surprising as already for $m = 2$ the rank of the associated subgroup can be arbitrarily high.

Proof of Theorem 5.8. This is a consequence of Theorem 5.6 and Lemma 5.7 using the embeddings $G \to G_{\text{am}}$ resp. $G \to G_{\text{HNN}}$. Note that in order to apply Lemma 5.7, we need an embedding of the free group F' of (5.2) and (5.3) into F_2 such that the image of F' is malnormal. Let k be the rank of F'. For example, the subgroup

$$\left\langle aba^{-1}b^{-1}, a^2b^2a^{-2}b^{-2}, \ldots, a^kb^ka^{-k}b^{-k} \right\rangle \leq F_2 = \langle a, b \rangle$$

is malnormal, see [BMR99, Ex. 1]; hence, there is such an embedding. □

5.2. Some Decidability Results

In the preceding section, we have seen that, in general, conjugacy is undecidable in fundamental groups of finite graphs of groups with free (abelian) vertex groups. In this section, we examine some special cases where conjugacy can be decided. In [BMR07b, BMR07a], Borovik, Myasnikov, and Remeslennikov showed that the conjugacy problem in HNN extensions and amalgamated products of f. g. free groups is decidable for elements of the so-called *regular part*. However, as in the Miller group it might be empty [BMR07c]; yet, in the Miller group conjugacy is decidable in P [BMR07c]. In [FMR10, MRF12], some conditions are given for the regular part being strongly generic. Hence, in these cases conjugacy is strongly generically decidable.

In this section, we present some special cases where the conjugacy problem in fundamental groups of finite graphs of groups with free and free abelian vertex groups is decidable – in some cases only on strongly generically sets and in other cases for all inputs.

5.2.1. Decidability with Free Vertex Groups

First, we examine conjugacy in fundamental groups of finite graphs of groups with f. g. free vertex groups and we restrict the edge groups further. We start by citing a result of Bogopolski, Martino, Maslakova, and Ventura for the case of one edge where both inclusions of the edge group in the vertex group are surjective. Recall that for two classes of groups $\mathcal{C}$ and $\mathcal{D}$, a group G is called $\mathcal{C}$-by-$\mathcal{D}$ if there is a normal subgroup N of G such that $N \in \mathcal{C}$ and $G/_N \in \mathcal{D}$.

Proposition 5.9 ([BMMV06])**.** *F.g.-free-by-cyclic groups have decidable conjugacy problem.*

In the case of cyclic edge groups, there are also some known results: In [Lip66], Lipschutz showed that an amalgamated product of f. g. free groups with cyclic amalgamated subgroup has decidable conjugacy problem. For HNN extension, a similar result holds: In [AS74], Anshel and Stebe showed that an HNN extension of a f. g. free group with cyclic associated subgroup has decidable conjugacy problem. Combining the results in [Ans76a, Ans76b] one obtains a solution for graphs of groups with one vertex and arbitrarily many edges. Here, we can generalize the results of [Ans76a, Ans76b, Lip66] to arbitrary underlying graphs – using similar techniques as in those papers. We also allow free abelian vertex groups.

Theorem 5.10. *Let $\mathcal{G}$ be a finite graph of groups and let all vertex groups be f. g. free or free abelian and all edge groups cyclic. Then the conjugacy problem of $\pi_1(\mathcal{G})$ is decidable.*

Proof. The word problem of $\pi_1(\mathcal{G})$ is decidable and Britton-reduced words can be computed by standard facts (see e. g. [KM02] or [LS01, Prop. I.2.21] for the subgroup membership problem in free groups and combine it with Lemma 3.15). Hence, by Lemma 3.18, we may assume that the input words are cyclically reduced. Using Stallings automata as presented in [KM02], one can decide whether some word can be conjugated into some cyclic subgroup inside a free vertex group – for abelian vertex groups this question is trivial – and in the positive case determine a conjugate element in the cyclic subgroup. Hence, case (i) and case (ii) of the Conjugacy Criterion, Theorem 3.19, can effectively be distinguished.

Case (i). We construct a new graph of groups $\mathcal{H}$ as follows. Let Y denote the underlying graph of the original graph of groups $\mathcal{G}$. We introduce an equivalence relation $\approx$ on its set of edges: For $x, y \in E(Y)$, we set $x \approx y$ if and only if $\iota(x) = \iota(y)$ and there are elements $a \in G_x^x$, $b \in G_y^y$ with $a \sim_{G_{\iota(x)}} b$ (i. e., $a =_{G_{\iota(x)}} b$ for abelian vertex groups). That means for every equivalence class $[x]$, we can choose a cyclic subgroup $\langle a_{[x]} \rangle \leq G_{\iota(x)}$ such that for all $y \in [x]$, the image of the edge group G_y^y can be conjugated into $\langle a_{[x]} \rangle$ inside $G_{\iota(x)}$. Let Y' be a graph with $V(Y') = (E(Y)/\approx) \cup V(Y)$ (note that $E(Y)/\approx$ is obtained from $V(Y)$ by "splitting" every vertex into several vertices, each of them corresponding to an equivalence classes of outgoing edges).

For $[y] \in E(Y)/\approx$, we assign as vertex group $H_{[y]} = \langle a_{[y]} \rangle$, and for $P \in V(Y)$, we assign the trivial group $H_P = \{1\}$. For every $y \in E(Y)$, we draw an edge from $[y]$ to $[\overline{y}]$ with the edge group $H_y = G_y = \langle a_y \rangle$ (hence, we assume $E(Y) \subseteq E(Y')$). The inclusion of $H_y = \langle a_y \rangle$ into $H_{\iota'(y)} = \langle a_{[y]} \rangle$ is given by $a_y \mapsto a_{[y]}^{e_y}$ where $a_{[y]}^{e_y}$ is the unique element in $\langle a_{[y]} \rangle$ which is conjugate to a_y^y inside $G_{\iota(y)}$ (note that for free vertex groups uniqueness follows immediately from the Conjugacy Criterion, Theorem 3.19). Moreover, for all $P \in V(Y)$, we draw an edge with trivial edge group from P to every $[y] \in E(Y)/\approx$ with $\iota(y) = P$.

Let $a \in G_x = H_x$ and $b \in G_y = H_y$ for some edges $x, y \in E(Y)$. By case (i) of Theorem 3.19, we have $a^x \sim_{F(\mathcal{G})} b^y$ if and only if there exists a sequence of elements $a^x = a_0, a_1, \ldots, a_m = b^y \in \bigcup_{y \in E(Y)} G_y^y$ such that for all i there is some $b_i \in E(Y) \cup \bigcup_{P \in V(Y)} (G_P \setminus \{1\})$ with $a_i = b_i a_{i-1} \bar{b}_i$. Now, by the construction of $\mathcal{H}$, this is the case if and only if there exists a sequence of elements $a^x = a'_0, a'_1, \ldots, a'_{m'} = b^y \in \bigcup_{y \in E(Y')} H_y^y$ such that for all i there is some $y_i \in E(Y')$ with $a_i = y_i a_{i-1} \overline{y}_i$ (note that with a little ambiguity, here, one time a^x denotes the inclusion into $G_{\iota(x)}$ and one time a^x is the inclusion into $\langle a_{[x]} \rangle = H_{\iota'(x)}$).

Hence, if $a \in G_x = H_x$ and $b \in G_y = H_y$ for some edges $x, y \in E(Y)$, then $a^x \sim_{F(\mathcal{G})} b^y$ if and only if $a^x \sim_{F(\mathcal{H})} b^y$. Now, we have reduced case (i) of Theorem 3.19 to conjugacy in a generalized Baumslag-Solitar group (see Chapter 7) and we can apply Theorem 7.23 below (note that in the proof of Theorem 7.23, we do not use any result of the present chapter).

Case (ii) is decidable as the conjugacy problem in free groups is decidable (see e. g. [LS01, Prop. I.2.14] or Corollary 5.15 below) and the conjugacy problem in free abelian groups is decidable.

Case (iii). Let $v = y_1 g_1 \cdots y_n g_n$ be a $\mathcal{G}$-factorization. W. l. o. g. (after some transposition) we may assume that $v \sim_{F(\mathcal{G})} w$ if and only if there is some $k \in \mathbb{Z}$ such that $c^k v c^{-k} =_{F(\mathcal{G})} w$ where c is the generator of the cyclic group $G_{y_1}^{y_1}$. Hence, we may assume that w is a $\mathcal{G}$-factorization of the form $y_1 h_1 \cdots y_n h_n$. Let $a_i \in G_{\iota(y_i)}$ and $b_i \in G_{\tau(y_i)}$ be the images of the generator of G_{y_i}, i. e., $G_{y_i}^{y_i} = \langle a_i \rangle$ and $G_{y_i}^{\overline{y}_i} = \langle b_i \rangle$. By Lemma 3.13 (iv), $v \sim_{F(\mathcal{G})} w$ if and only if there are numbers $k_1, \ldots, k_n \in \mathbb{Z}$ with

$$\begin{aligned} b_i^{k_i} g_i a_{i+1}^{-k_{i+1}} &= h_i && \text{for } 1 \leq i < n, \\ b_n^{k_n} g_n a_1^{-k_1} &= h_n. \end{aligned}$$

For each i, the existence of numbers $k_i, k_{i+1} \in \mathbb{Z}$ with $b_i^{k_i} g_i a_{i+1}^{-k_{i+1}} = h_i$ can be determined by standard facts about free groups [AM84, Thm. 3.4], see also [KM02] or the *cardinality search problem* in [BMR07b]. (For free abelian vertex groups, it is straightforward to determine such $k_i, k_{i+1} \in \mathbb{Z}$.) Moreover, if there is more than one solution for some i, then there are integers α_i, β_i such that all solutions for this i are of the form $(k_i + \ell\alpha_i, k_{i+1} + \ell\beta_i)$ for $\ell \in \mathbb{Z}$ where (k_i, k_{i+1}) is one arbitrary solution. These α_i, β_i can also be determined effectively. Hence, for deciding conjugacy, it remains to solve a system of linear integer equations. □

Remark 5.11. A closer analysis of the word problem leads to a polynomial time bound in Theorem 5.10. The main idea is to represent elements of the edge groups as binary integers. However, we do not present a proof because we would have to introduce Stallings automata.

Also, we want to emphasize that Theorem 5.10 can be generalized to a much wider class of groups comprising right-angled Artin groups (sometimes called graph groups). The key point is that two elements in some edge group are conjugate inside some vertex group if and only if they are equal, compare to [Ans76b] (edge groups malnormal in vertex groups) or [Loc92] (edge groups central in vertex groups).

5.2.2. Decidability with Free Abelian Vertex Groups

As a direct consequence of Theorem 5.10, we obtain:

Corollary 5.12. *Let $\mathcal{G}$ be a finite graph of groups and let all vertex groups be f. g. free abelian groups and all edge groups cyclic. Then the conjugacy problem of $\pi_1(\mathcal{G})$ decidable.*

The following result is a stronger version of [Bee11b, Cor. 5.4.4] where only decidability is proven.

Proposition 5.13. *Let $\mathcal{G}$ be a finite graph of groups with underlying graph Y. Furthermore, let the vertex groups all be f. g. free abelian. Then the conjugacy problem of $G = \pi_1(\mathcal{G})$ is in P for elements which cannot be conjugated into any vertex group.*

Proof. The proof relies on the result by Frumkin [Fru77] and von zur Gathen and Sieveking [vzGS78] that the existence of an integer solution of a system of linear equations can be checked in polynomial time, and if there is a solution, it can be computed in polynomial time, see also [Sch86, Cor. 5.3b].

We represent all elements of the free abelian vertex and edge groups as binary integers (more precisely, as vectors of binary integers). Obviously, the word problems of all vertex groups are in P. The inclusions $G_y \to G_{\iota(y)}$ and $G_y \to G_{\tau(y)}$ are represented by matrices A_y and B_y. (Note that these matrices need not to be square matrices; in particular, they need not to be invertible.) Hence, the subgroup membership problem of the edge groups in the vertex groups is in P by the result of [Fru77, vzGS78], and also the isomorphisms $G_y^y \to G_{\overline{y}}^{\overline{y}}$ are computable in polynomial time for all $y \in E(Y)$. Thus, by Lemma 3.15, words can be effectively Britton-reduced, and by Lemma 3.18, this allows us to reduce words cyclically. Moreover, in every application of a Britton reduction the sizes of the binary representations of the occurring numbers increase only by a constant. Hence, the algorithm in Lemma 3.15 runs in polynomial time. By assumption, we are in case (iii) of the Conjugacy Criterion, Theorem 3.19. Let

$$v = y_1 g_1 \cdots y_n g_n, \qquad w = y_1 h_1 \cdots y_n h_n$$

be cyclically reduced $\mathcal{G}$-factorizations with g_i, h_i vectors of binary integers for all i (by (iii) of Theorem 3.19, we may assume that the edge sequences of v and w are the same).

W. l. o. g. $v \sim_G w$ if and only if there is some $a \in G_{\iota(y_1)}$ such that $ava^{-1} =_G w$. By Lemma 3.13 (iv), it follows – as in the proof of Theorem 5.10 – that $ava^{-1} =_G w$ if and only if there are vectors $a_1, \dots, a_n$ with $a_i \in G_{y_i}$ such that $a = A_{y_1} a_1$ and

$$\begin{aligned} B_{y_i} a_i - A_{y_{i+1}} a_{i+1} + g_i &= h_i && \text{for } 1 \leq i < n, \\ B_{y_n} a_n - A_{y_1} a_1 + g_n &= h_n. \end{aligned}$$

Note that here $+$ is the binary group operation of H, and as before, A_y and B_y are the matrices representing the inclusions $G_y \to G_{\iota(y)}$ and $G_y \to G_{\tau(y)}$. This is a system of linear integer equations of size linear in the input size; hence, the existence of an integer solution is decidable in polynomial time by [Fru77, vzGS78]. □

Remark 5.14. In [Bee11b, Thm. 5.8.1], Beeker proved also that the multiple conjugacy problem in fundamental groups of finite graphs of groups with f. g. free abelian vertex groups is decidable for tuples which generate subgroups which cannot be conjugated into any vertex group. Following the proof of Proposition 5.13, this more general problem can also be shown to be in P. The multiple conjugacy problem of a group G is the following problem: Given two tuples $(v_1, \dots, v_k)$ and $(w_1, \dots, w_k)$ of input words, decide whether there exists some $z \in G$ such that $zv_iz^{-1} =_G w_i$ for all i.

As in [Bee11b], w. l. o. g. we may assume that v_1 and w_1 cannot be conjugated into any vertex group. As a first step, v_1 and w_1 are cyclically reduced like in the proof of Proposition 5.13 – all necessary conjugations are also applied to v_i resp. w_i for $2 \leq i \leq k$. Next, for all cyclic permutations of v_1, the respective conjugations are also applied to v_i for $2 \leq i \leq k$. Then Britton reductions are applied to all words. As a final step, it remains to solve a system of linear equations like at the end of the proof of Proposition 5.13.

Corollary 5.15. *Let $\mathcal{G}$ be a finite graph of groups with all vertex groups f. g. free abelian and let the underlying graph Y be a tree. Then the conjugacy problem of $\pi_1(\mathcal{G})$ is in* P.

Proof. By Proposition 5.13, we only need to consider case (i) and (ii) of Theorem 3.19. Let $v \in G_P$ and $w \in G_Q$ for some $P, Q \in V(Y)$ with $v \sim_{F(\mathcal{G})} w$ and let $z \in \Pi(\mathcal{G}, P, Q)$ be some shortest word with $v =_{F(\mathcal{G})} zw\bar{z}$. As all the vertex groups are abelian, z contains only letters from $E(Y)$. Now, Y is a tree; thus, there exists exactly one path without back-tracking from P to Q. Hence, there is at most one possible choice for z (depending only on P and Q). Therefore, it suffices to solve the word problem to check whether $v =_{F(\mathcal{G})} zw\bar{z}$ for this unique z – by the proof of Proposition 5.13 this is in P. □

Corollary 5.16. *Let $\mathcal{G}$ be a finite graph of groups. If all the vertex groups are f. g. free abelian, then the conjugacy problem of $\pi_1(\mathcal{G})$ is doubly generically in* P. *If moreover*

- *$\mathcal{G}$ is reduced and has at least two undirected edges, or*
- *$\mathcal{G}$ has one undirected edge $\{y, \bar{y}\}$ with $[G_{\iota(y)} : G_y^y] \geq 3$ and $[G_{\iota(\bar{y})} : G_{\bar{y}}^{\bar{y}}] \geq 2$, or*
- *$\mathcal{G}$ has one undirected edge $\{y, \bar{y}\}$ which is a loop and $[G_{\iota(y)} : G_y^y], [G_{\iota(\bar{y})} : G_{\bar{y}}^{\bar{y}}] \geq 2$,*

then the conjugacy problem of $\pi_1(\mathcal{G})$ is doubly strongly generically in P.

Proof. If the graph of groups has no edges, then conjugacy can be decided in P. Let T be a spanning tree of Y. By Remark 4.20 and its analog for amalgamated products, the set $\left\{w \in \Sigma^* \;\middle|\; \forall z \in \pi_1(\mathcal{G},T) : zwz^{-1} \notin_{\pi_1(\mathcal{G},T)} G_P\right\}$ is generic in Σ^* for every $P \in V(Y)$. If one of the stronger conditions holds, then, by Theorem 4.17 (if the underlying graph is a tree) resp. Theorem 4.18 (otherwise) together with Theorem 4.3, the set $\left\{w \in \Sigma^* \;\middle|\; \forall z \in \pi_1(\mathcal{G},T) : zwz^{-1} \notin_{\pi_1(\mathcal{G},T)} G_P\right\}$ is strongly generic in Σ^* for every $P \in V(Y)$. Hence, the result follows from Proposition 5.13. □

Corollary 5.17. *F.g.-free-abelian-by-free groups have conjugacy problem doubly strongly generically in* P.

Proof. By Corollary 5.16, it remains to consider f.g.-free-abelian-by-cyclic groups. In this case the conjugacy problem is in P by [CK14]. □

The next result generalizes [BMV10, Cor. 6.8] and [CK14] which state that f.g.-free-abelian-by-cyclic groups have decidable conjugacy problem (resp. conjugacy problem in P).

Proposition 5.18. *Let H be a f. g. free abelian group, $A, B \leq H$, $\varphi : A \to B$ an isomorphism, and $G = \langle H, t \mid tat^{-1} = \varphi(a)$ for $a \in A \rangle$ an HNN extension. Then the conjugacy problem of G is decidable.*

Proof. As in the proof of Proposition 5.13, words can be reduced cyclically. By Proposition 5.13, we need to consider cases (i) and (ii) of the Conjugacy Criterion, Theorem 3.19, only. Hence, let $v, w \in H$. In this case, the conjugation with some element of H has no effect. Hence, conjugation in G reduces to the question whether there is some $k \in \mathbb{Z}$ with $\varphi^k(v) =_G w$. We reduce this question to the *orbit problem* for rational matrices, which has been shown to be decidable by Kannan and Lipton [KL86] (preliminary version in [KL80]).

As H is a f. g. free abelian group, it embeds canonically into some vector space $\mathbb{Q}^n$ and φ induces an isomorphism between subspaces $\tilde{A}$ and $\tilde{B}$ of $\mathbb{Q}^n$. We choose a basis of $\tilde{A}$ and extend it to a basis R of $\mathbb{Q}^n$. Now, we set $\tilde{\varphi}(u) = \varphi(u)$ for $u \in R \cap \tilde{A}$, and $\tilde{\varphi}(u) = 0$ for $u \in R \setminus \tilde{A}$.

It is clear that $\varphi^k(v) = w$ for some $k \geq 0$ implies that $\tilde{\varphi}^k(v) = w$ for the same k. The latter can be checked using the algorithm of [KL86]. Moreover, if there is such k, then the smallest such k can be computed by trying all values. Now, if $\tilde{\varphi}^k(v) = w$, then as a final step it can be checked whether $\tilde{\varphi}^i(v) \in \mathrm{dom}(\varphi)$ for all $0 \leq i < k$. If so, then we have $\varphi^k(v) = w$. With the same procedure, we can check whether there is some $k \leq 0$ with $\varphi^k(v) = w$. Hence, we are done. □

We conjecture that the problem of Proposition 5.18 can be decided in polynomial time, indeed. For proving such a result, it remains to find a polynomial time algorithm for the check whether $\tilde{\varphi}^i(v) \in \mathrm{dom}(\varphi)$ for all $0 \leq i < k$, because the algorithm of [KL86] runs in polynomial time.

Note that a polynomial time bound would also imply that the statement of Corollary 5.16 holds in all cases with "doubly strongly generically in P".

Chapter 6.

Finite Edge Groups

In this chapter, we want to describe how to solve the conjugacy problem in fundamental groups of finite graphs of groups with finite edge groups. In [Hor89, Loc93], it has been shown that if all edge groups are finite cyclic and the conjugacy problem in the vertex groups is decidable, then also the conjugacy problem in the fundamental group of the graph of groups is (uniformly) decidable. Here, we examine what can be stated about the complexity of the word and conjugacy problem of the fundamental group if we know the complexity of the word and conjugacy problem of all vertex groups. We start by introducing some notation and stating some straightforward observations. Then we examine the time complexity more carefully under the assumption of a RAM model. Finally, we analyze the circuit complexity of conjugacy.

In the following, $\mathcal{C}$ denotes some arbitrary complexity class. Recall that $\mathsf{P}^{\mathcal{C}}$ denotes the class of problems decidable in polynomial time with oracles for some finite set of languages in $\mathcal{C}$.

Let $\mathcal{G}$ be a graph of groups with finite edge groups and underlying finite graph Y such that $\pi_1(\mathcal{G})$ is finitely generated. By [Dic80, Cor. II.3.4], we know that all the vertex groups are finitely generated. For $P \in V(Y)$, let Σ_P be a finite symmetric generating set of G_P such that $G_y^y \setminus \{1\} \subseteq \Sigma_P$ for all $y \in E(Y)$ with $\iota(y) = P$, i. e., the edge groups without the identity are a subset of the alphabet. We obtain a finite symmetric generating set

$$\Sigma = E(Y) \cup \bigcup_{P \in V(Y)} \Sigma_P$$

for $F(\mathcal{G})$ (where the union is disjoint). As in Section 3.3, we write $\mathcal{A} = \bigcup_{y \in E(Y)} G_y^y$. In particular, we have $\mathcal{A} \setminus \{1\} \subseteq \Sigma$.

Assume that we can solve the word problem in all vertex groups. Then also for every edge $y \in E(Y)$ the subgroup membership problem of the edge group G_y^y in the vertex group $G_{\iota(y)}$ is decidable: on input $w \in \Sigma_{\iota(y)}$, one simply checks for each $a \in G_y$ whether $w =_{G_{\iota(y)}} a^y$. Hence, also the isomorphisms $G_y^y \to G_{\bar{y}}^{\bar{y}}$ are effectively computable. Thus, by Lemma 3.15, the word problem of $\pi_1(\mathcal{G})$ is decidable, and words can be effectively Britton-reduced. By Lemma 3.18, this implies that words can be effectively cyclically reduced. Furthermore, every application of a Britton reduction to some word decreases its length because we assumed $\mathcal{A} \subseteq \Sigma$. Hence, if the input word is of length N, then the algorithms for the word problem of the groups G_P are executed on word of length at most N. Moreover, at most $\mathcal{O}(N^2)$ such calls are performed. Thus, we have the following fact, which is also a direct consequence of [Tre88, Thm. 1.12]:

Proposition 6.1 ([Tre88])**.** *Assume that* $\mathrm{WP}(G_P) \in \mathcal{C}$ *for all* $P \in V(Y)$*. Then the word problem of* $\pi_1(\mathcal{G})$ *is in* $\mathsf{P}^{\mathcal{C}}$*. Furthermore, on input of a* $\mathcal{G}$*-factorization* $w \in \Sigma^*$*, a cyclically reduced* $\mathcal{G}$*-factorization* $\hat{w} \in \Sigma^*$ *with* $\hat{w} \sim_{F(\mathcal{G})} w$ *can be computed in* $\mathsf{P}^{\mathcal{C}}$*.*

Note that in [HL09, HL11] Haubold and Lohrey show that Proposition 6.1 is true even when the input words are given in compressed form.

Now, consider two cyclically reduced $\mathcal{G}$-factorizations

$$v = x_1 g_1 \cdots x_m g_m, \qquad w = y_1 h_1 \cdots y_n h_n$$

with $m, n > 0$ and $g_i \in \Sigma^*_{\tau(x_i)}$, $h_i \in \Sigma^*_{\tau(y_i)}$ for all i. If $v \sim_{F(\mathcal{G})} w$, then we know by the Conjugacy Criterion, Theorem 3.19 (iii), that $m = n$ and there is a cyclic permutation w' of w and some $a \in \mathcal{A}$ such that $v =_{F(\mathcal{G})} aw'a^{-1}$. There are exactly n cyclic permutations and only a finite constant number of $a \in \mathcal{A}$. Hence, in this case, the conjugacy problem is decidable, and more precisely, we have:

Proposition 6.2. *Let the word problem of* G_P *be in* $\mathcal{C}$ *for all* $P \in V(Y)$*. Then for* $\mathcal{G}$*-factorizations* $v, w \in \Sigma^*$ *which in* $F(\mathcal{G})$ *cannot be conjugated into* G_P *for any* $P \in V(Y)$*, it can be decided whether* $v \sim_{F(\mathcal{G})} w$ *in* $\mathsf{P}^{\mathcal{C}}$*.*

Now, we can apply the results about amenability and strongly generic sets, Theorem 4.21, to our setting. Recall that by Stallings' structure theorem [Sta71] a f. g. group has infinitely many ends if and only if it splits as an HNN extension $G = \langle H, t \mid tat^{-1} = \varphi(a) \text{ for } a \in A \rangle$ with $A, \varphi(A) \leq H$ finite and $[H : A]$, $[H : \varphi(A)] \geq 2$ or as an amalgamated product $G = G_1 *_A G_2$ with A finite, $[G_1 : A] \geq 3$, and $[G_2 : A] \geq 2$.

Corollary 6.3. *Let* G *be a finitely generated group with infinitely many ends. If the word problem of* G *is in* $\mathcal{C}$*, then the conjugacy problem of* G *is doubly strongly generically in* $\mathsf{P}^{\mathcal{C}}$*. In particular, if the word problem of* G *is in* P*, then the conjugacy problem of* G *is doubly strongly generically in* P*.*

We have not considered cases (i) and (ii) of Theorem 3.19, yet. Hence, let $v \in G_P$, $w \in G_Q$. By Theorem 3.19, we have $v \sim_{F(\mathcal{G})} w$ if and only if either $P = Q$ and $v \sim_{G_P} w$ or there are elements $a_v, a_w \in \mathcal{A}$ with $v \sim_{G_P} a_v$, $w \sim_{G_Q} a_w$ and $a_v \sim_{F(\mathcal{G})} a_w$. Since, $\mathcal{A}$ is finite, in both cases it can be checked whether $v \sim_{F(\mathcal{G})} w$ with at most $2\,|\mathcal{A}| + 1$ calls to the conjugacy problem for G_P and G_Q.

Corollary 6.4. *Let* $\mathcal{G}$ *be a finite graph of groups with finite edge groups. If for every* $P \in V(Y)$ *the conjugacy problem of the vertex group* G_P *is in* $\mathcal{C}$*, then the conjugacy problem of* $\pi_1(\mathcal{G})$ *is in* $\mathsf{P}^{\mathcal{C}}$*.*

Hence, if the conjugacy problem in the vertex groups is decidable in polynomial (resp. exponential, elementary) time, then the conjugacy problem in the fundamental group of the finite graph of groups with finite edge groups is decidable in polynomial (resp. exponential, elementary) time.

6.1. Complexity on a Random Access Machine

In this section, we want to examine the time complexity of the conjugacy problem in fundamental groups of finite graphs of groups with finite edge groups more carefully under the assumption of a RAM model for the complexity (for the definition see Appendix A). The setting is that we only assume a bound on the complexity of conjugacy in the vertex groups. We treat the vertex groups as black boxes and do not require any additional algorithmic properties; in particular, we do *not* assume that normal forms can be computed efficiently for the vertex groups. Under this assumptions we aim for an algorithm for conjugacy in the fundamental group which provides a good bound for the running time.

We keep all the notation as before (in particular, $\mathcal{A} \subseteq \Sigma$, and both sets are finite). For all $P \in V(Y)$, we assume that we can solve the conjugacy problem in time $T(N)$ on a RAM (where the input consists of two words $v, w \in \Sigma_P^*$ with $|v| + |w| = N$). We assume T to be a function on the real numbers in order to simplify the analysis. For the function $T : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, we have the following requirement (beside being not a constant zero function):

$$\sum_{i=0}^{k} T(N_i) \leq T\left(\sum_{i=0}^{k} N_i\right) \text{ for all } k \in \mathbb{N}, N_i \in \mathbb{R}_{\geq 0}.$$

This is a very natural condition and not very restricting. In particular, it implies that T is monotone, that $T(0) = 0$, and that $c \cdot T(N) \leq T(cN)$ for $c \geq 1$. At some places, we want T to meet an additional condition. We say T meets condition $(*)$, if there is some $\epsilon > 0$ such that

$$T(cN) \geq c^{1+\epsilon} \cdot T(N) \qquad \text{for all } c > 1, N \in \mathbb{N}.$$

If $w = h_0 y_1 h_1 \cdots y_n h_n \in \Sigma^*$ is a $\mathcal{G}$-factorization with $|w| = N$ where $h_i \in \Sigma^*_{\tau(y_i)}$. The first task is to compute a Britton-reduced $\mathcal{G}$-factorization $\hat{w} \in \Sigma^*$ with $\hat{w} =_{F(\mathcal{G})} w$. As we have already seen, we obtain a bound for the running time of $n \cdot T(N)$ if we use the straightforward algorithm and perform Britton reductions from left to right, i. e., if we check always for the left-most factor $yh\overline{y}$ with $h \in \Sigma^*_{\tau(y)}$ whether $h \in G_y^{\overline{y}}$. The following example shows that this is essentially the best bound we can obtain for the left-to-right algorithm.

Example 6.5. Let $w = yh(\overline{y}y)^{k-2}h'\overline{y}$ with $h, h' \in \Sigma^*_{\tau(y)}$ such that $|h| \geq |h'| > k$, $h \notin G_y^{\overline{y}}$ and $hh' =_{G_{\tau(y)}} 1$. Then, $|h| \geq N/3$ for $N = |w|$. The left-to-right algorithm would test $k-2$ times whether $h \in G_y^{\overline{y}}$; hence, it needs $\Omega(k \cdot T(N/3))$ time if the algorithm for the word problem of $G_{\tau(y)}$ runs in time $\Theta(T(N))$.

However, we can do better if we choose the right ordering for the tests whether a Britton reduction can be performed. The key idea is that the algorithm does not work from left to right, but checks first for the shortest candidates $yh\overline{y}$ with $h \in \Sigma^*_{\tau(y)}$ whether $h \in G_y^{\overline{y}}$. More precisely, the algorithm to Britton-reduce a given $\mathcal{G}$-factorization $w \in \Sigma^*$ with $|w| = N$ and $\mathcal{G}$-length n works as follows (for pseudocode see Algorithm 6.1):

Algorithm 6.1 *BrittonReduce*

procedure BrittonReduce($w \in \Sigma^*$)
begin
  priorityqueue $Q \leftarrow \emptyset$
  scan w for factors $yh\overline{y}$ with $y \in E(Y)$ and $h \in \Sigma^*_{\tau(y)}$
  add all these factors h to Q (the weight is defined by $|h|$)
  while ($Q \neq \emptyset$) **do**
    $h \leftarrow$ deletemin(Q)
    if ($h = a^{\overline{y}} \in G_y^{\overline{y}}$) ($*$ can be checked in $|\mathcal{A}| \cdot T(|h|+1)$ time $*$)
      replace $ya^{\overline{y}}\overline{y}$ by a^y in w
      from the current position scan w forward and backward whether
        a factor $y'h'\overline{y}'$ for some $y' \in E(Y)$ and $h' \in \Sigma^*_{\tau(y')}$ has been produced
      if (there is a new factor $y'h'\overline{y}'$ in w) ($*$ there is at most one such factor $*$)
        insert h' into Q
      endif
    endif
  endwhile
end

We use a priority queue Q (e. g. a heap, see [CLRS09]) which stores words over Σ ordered by their length. In a first initialization phase, the input word w is scanned from left to right, and for every factor $yh\overline{y}$ with $h \in \Sigma^*_{\tau(y)}$ (in particular, h does not contain any letter $y \in E(Y)$), the word h is added to Q (more precisely, the index of the position in w and the length are stored to have a fast access and not to have to move the data around – however, for simplicity, we assume that also h is stored in the priority queue). The next phase of the algorithm modifies the input w until it remains Britton-reduced. In order to do so, the following loop is repeated until Q is empty. Here, $w^{(t-1)}$ denotes the modified input before the t-th iteration, i. e., $w^{(0)} = w$.

First, the shortest element is removed from Q. Say it is h and $w^{(t-1)} = w_1 yh\overline{y}w_2$ for some $w_1, w_2 \in \Sigma^*$, $y \in E(Y)$. Now, the check whether $h \in G_y^{\overline{y}}$ is performed using at most $|G_y|$ calls to the word problem in $G_{\tau(y)}$. If $h \notin G_y^{\overline{y}}$, then $w^{(t)} = w^{(t-1)}$ and nothing else happens. If $h \in G_y^{\overline{y}}$, then let $a \in G_y$ with $h =_{G_{\tau(y)}} a^{\overline{y}}$. We set $w^{(t)} = w_1 a^y w_2$. In the case that a new factor $y'h'\overline{y}'$ with $h' \in \Sigma^*_{\tau(y')}$ has been created, we add h' to Q – this can be checked by scanning from the position of h forward and backward until reaching the next letter $y' \in E(Y)$.

At the end of the algorithm, the word $w^{(t)}$ does not have a factor $yh\overline{y}$ with $h \in G_y^{\overline{y}}$ for any $y \in E(Y)$, and hence it is Britton-reduced. Thus, the algorithm is correct. The key point is that, due to the special ordering of the Britton reductions, the time spent for unsuccessful tests ($h \notin G_y^{\overline{y}}$) can be minimized. The following observations are easy to see, but crucial for the further analysis (recall that $N = |w|$, $n = |w|_{\mathcal{G}}$):

(i) At any time t during the execution of Algorithm 6.1, we have $\left|w^{(t)}\right| \leq N$.

(ii) Let l denote the length of the shortest element in the priority queue Q. And $|Q|$ the number of elements in Q. At every point during the execution of Algorithm 6.1, we have

$$l \cdot |Q| \leq N - n.$$

(iii) During the execution of the second phase of Algorithm 6.1, the size of Q never increases; after every unsuccessful test whether $h \in G_y^{\bar{y}}$ for some $h \in Q$, the size of Q decreases. In particular, there occur at most n unsuccessful tests.

Let $H(N) = \sum_{k=1}^{N} \frac{1}{k} \in \Theta(\log N)$ denote the harmonic function.

Lemma 6.6. *During the execution of Algorithm* 6.1*, all unsuccessful tests whether $h \in G_y^{\bar{y}}$ for some $h \in \Sigma^*_{\tau(y)}$ together cost as most $|\mathcal{A}| \cdot H(n) \cdot T(N)$ time. Moreover, if T meets condition $(*)$, then all the unsuccessful tests together cost at most $\mathcal{O}(|\mathcal{A}| \cdot T(N))$ time.*

Proof. One test whether $h \in G_y^{\bar{y}}$ uses at most $|\mathcal{A}| \cdot T(|h| + 1)$ time. Let $T_{\mathrm{u}}(w)$ denote the time consumed by all unsuccessful tests together and let h_k be the element which is unsuccessfully tested at the time when $|Q| = k$ (by (iii) we know that there is at most one such word – if there is none, we simply assume h_k is the empty word). Hence, we have

$$T_{\mathrm{u}}(w) \leq \sum_{k=1}^{n} |\mathcal{A}| \cdot T(|h_k| + 1).$$

By (ii) and our assumptions on T, we obtain

$$\begin{aligned} &\leq |\mathcal{A}| \cdot \sum_{k=1}^{n} T\left(\frac{N-n}{k} + 1\right) \\ &\leq |\mathcal{A}| \cdot \sum_{k=1}^{n} \frac{1}{k} \cdot T\left(N\right) \\ &= |\mathcal{A}| \cdot H(n) \cdot T(N). \end{aligned}$$

If $(*)$ holds, we can do better:

$$\begin{aligned} T_{\mathrm{u}}(n) \leq\ & |\mathcal{A}| \cdot \sum_{k=1}^{n} T\left(\frac{N-n}{k} + 1\right) \\ \leq\ & |\mathcal{A}| \cdot \sum_{k=1}^{n} T\left(\frac{N}{k}\right) \\ \leq\ & |\mathcal{A}| \cdot \sum_{k=1}^{n} \frac{1}{k^{1+\epsilon}} T\left(N\right) \\ \leq\ & |\mathcal{A}| \cdot T\left(N\right) \cdot \sum_{k>0} \frac{1}{k^{1+\epsilon}}. \end{aligned}$$

□

Lemma 6.7. *During the execution of Algorithm* 6.1*, all successful tests whether* $h \in G_y^{\overline{y}}$ *for* $h \in \Sigma^*_{\tau(y)}$ *together consume at most* $|\mathcal{A}| \cdot T(N)$ *time.*

Proof. Everytime a successful test is performed on some $h \in \Sigma^*_{\tau(y)}$, $yh\overline{y}$ is replaced by a single letter. So during its "lifetime" every letter is tested successfully only once. Let s denote the number of successful tests. We have $s \leq n/2$. The total number of letters added is bounded by s since only after successful tests letters are added. At the beginning, there are $N - n$ letters from $\Sigma \setminus E(Y)$ in w (letters from $E(Y)$ are never included in any test). Thus, the total number of letters which are part of the input of successful tests is bounded by $N - n + s \leq N$. Hence, by our assumption on T, we conclude that the total time for successful tests is bounded by $|\mathcal{A}| \cdot T(N)$. □

Proposition 6.8. *Algorithm* 6.1 *takes a* $\mathcal{G}$*-factorization* $w \in \Sigma^*$ *as input and computes a Britton-reduced* $\mathcal{G}$*-factorization* $\hat{w}$ *with* $\hat{w} =_{F(\mathcal{G})} w$ *in time* $\mathcal{O}(|\mathcal{A}| \cdot \log n \cdot T(N))$*, where* $\mathcal{A}$ *and* T *are as above. If furthermore* $(*)$ *holds, then the algorithm runs in time* $\mathcal{O}(|\mathcal{A}| \cdot T(N))$*.*

Proof. By Lemma 6.6 and Lemma 6.7, at most $\mathcal{O}(|\mathcal{A}| \cdot \log n \cdot T(N))$ (resp. $\mathcal{O}(|\mathcal{A}| \cdot T(N))$) time is spent in the calls for the word problem of the vertex groups. All other operations (scanning for factors $yh\overline{y}$, insertion into the priority queue, removal from the priority queue) are bounded by $\mathcal{O}(N + n \log n)$. If $(*)$ holds, we have $n \log n \in O(T(n))$. □

Note that the factor $|\mathcal{A}|$ is only a rough upper bound and could be replaced by $\max\{|G_y| \mid y \in E(Y)\}$. However, for simplicity, we stick to $|\mathcal{A}|$.

Remark 6.9. If $v, w \in \Sigma^*$ are Britton-reduced $\mathcal{G}$-factorizations with $|v| + |w| = N$, then the question whether $v =_{F(\mathcal{G})} w$ is decidable in time $\mathcal{O}(|\mathcal{A}| \cdot T(N))$ – even without $(*)$. In fact, this is easy to see because the only place where Britton reductions might occur in $v\overline{w}$ is the place where v and $\overline{w}$ meet. Hence, we do not need the priority queue, and all but one tests whether $h \in G_y^{\overline{y}}$ for some $h \in \Sigma^*_{\tau(y)}$ are successful. The length of $v\overline{w}$ is by assumption equal to N. Hence, the result follows as in the proof of Lemma 6.7.

Applying Lemma 3.18 leads to:

Corollary 6.10. *There is an algorithm which on input of a* $\mathcal{G}$*-factorization* $w \in \Sigma^*$*, computes a cyclically reduced* $\mathcal{G}$*-factorization* $\hat{w}$ *with* $\hat{w} \sim_{F(\mathcal{G})} w$ *in time* $\mathcal{O}(|\mathcal{A}| \cdot \log n \cdot T(N))$*, where* n*,* N*,* $\mathcal{A}$ *and* T *are as above. If furthermore* $(*)$ *holds, then the same algorithm runs in time* $\mathcal{O}(|\mathcal{A}| \cdot T(N))$*.*

Now, we are able to reduce words cyclically. The next step is to test whether two cyclically reduced words are conjugate. By the Conjugacy Criterion, Theorem 3.19, we have to distinguish three cases.

First consider case (i) and (ii) of Theorem 3.19. Hence, let $v \in \Sigma_P^*$, $w \in \Sigma_Q^*$. As we have seen before Corollary 6.4, $v \sim_{F(\mathcal{G})} w$ can be decided with at most $2|\mathcal{A}| + 1$ calls to the conjugacy problem of G_P and G_Q.

For case (iii) of Theorem 3.19, the naive algorithm is to test for every $a \in \mathcal{A}$ and every cyclic permutation v' of v whether $awa^{-1} =_{F(\mathcal{G})} v'$. This leads to a running time

in $\mathcal{O}(|\mathcal{A}|^2 \cdot n \cdot T(N))$. However, we can do better.

A crucial tool for the improved algorithm is the string matching algorithm by Knuth, Morris, and Pratt [KMP77] (which has also been described by Matiyasevich [Mat73]). This algorithm takes two strings $v, w \in \Sigma^*$ over some finite or infinite alphabet Σ as input and gives as output all occurrences of v as substring of w, i. e., all factorizations $w = u_1 v u_2$ with $u_1, u_2 \in \Sigma^*$ (more precisely, the output is a list of all indices where the factors v start). In order to do so, the algorithm performs $\mathcal{O}(N)$ comparisons between letters and $\mathcal{O}(N)$ other operations, where $N = |v| + |w|$.

One idea is to use the Knuth-Morris-Pratt algorithm over the (possibly infinite) alphabet $\Delta = E(Y) \cup \bigcup_{P \in V(Y)} (G_P \setminus \{1\})$. Where the comparisons between letters (i. e., group elements of vertex groups) are made by calls to the word problem of the G_P. However, this might lead to a running time of $\Omega(n \cdot T(N))$: Although the Knuth-Morris-Pratt algorithm gives an $\mathcal{O}(n)$ bound for the total number of comparisons (where n is as before the $\mathcal{G}$-length of the input), there might be elements which are compared up to $\Omega(n)$ times. If elements in Δ are represented as words over Σ, then there might be some element of length $\Omega(N)$ which is compared $\Omega(n)$ times. If $n \in \Omega(N)$, this gives $\Omega(N \cdot T(N))$ running time. Hence, this approach does not improve over the trivial algorithm.

Another idea is to assume that some normal forms of elements in the vertex groups can be computed in time $T(N)$. Then one could apply the Knuth-Morris-Pratt algorithm on these normal forms leading to $\mathcal{O}(T(N))$ running time for conjugacy. However, our aim is to not make any additional assumptions on the vertex groups; hence, we content ourselves with Theorem 6.11:

Theorem 6.11. *Let $\mathcal{G}$ be a finite graph of groups with finite edge groups and assume that the word problem of the vertex groups G_P is decidable in time $T(k)$ for all $P \in V(Y)$, where k is the length of the input word. Then there is an algorithm running in time $\mathcal{O}(|\mathcal{A}|^3 \cdot \log n \cdot T(4N))$ for the following problem: on input of two cyclically reduced $\mathcal{G}$-factorizations $v, w \in \Sigma^*$ with $|v|$, $|w| \leq N$ and $|v|_{\mathcal{G}} = |w|_{\mathcal{G}} = n > 0$, decide whether $v \sim_{F(\mathcal{G})} w$.*

Note that with a more careful analysis of the running time in the proof of Theorem 6.11, we could replace $T(4N)$ by $T(2N)$ in the statement of Theorem 6.11. As we are principally interested in functions T which are bounded by a polynomial, we content ourselves with $T(4N)$.

Proof. Let $v = x_1 g_1 \cdots x_n g_n$ and $w = y_1 h_1 \cdots y_n h_n$ be cyclically reduced $\mathcal{G}$-factorizations. We need to check whether there is some cyclic permutation v' of v and some $a \in \mathcal{A}$ with $v' =_{F(\mathcal{G})} awa^{-1}$. In order to do so, we consider a slightly more general situation and introduce the following notation:

Let $k \in \mathbb{N}$. With $\mathcal{V}_k \subseteq (\Sigma^*)^k$ we denote the set of all k-tuples $\vec{v} = (v_1, \ldots, v_k)$ where v_i is Britton-reduced and starts with some $y_i \in E(Y)$ for all $1 \leq i \leq k$. We set $G_i^{\vec{v}} = G_{y_i}^{y_i}$ and $|\vec{v}| = \sum_{i=1}^k |v_i|$. Also, for $v, w \in (\Sigma^*)^k$, we write $\vec{v} =_{F(\mathcal{G})} \vec{w}$ if $v_1 \cdots v_k =_{F(\mathcal{G})} w_1 \cdots w_k$, and we write $\vec{v} = \vec{w}$ if all components are equal (as words). A *cyclic permutation* of

$\vec{v} = (v_1, \ldots, v_k) \in \mathcal{V}_k$ is a tuple $\vec{v}' = (v_i, \ldots, v_k, v_1, \ldots, v_{i-1})$ for some i. We define an equivalence relation $\approx$ over $\mathcal{V}_k$ by

$$\vec{v} \approx \vec{w} \text{ if and only if for all } i \text{ there are } a_i \in G_i^{\vec{v}}, b_i \in G_{i+1}^{\vec{v}} \text{ with } a_i v_i b_i =_{F(\mathcal{G})} w_i$$

for $\vec{v} = (v_1, \ldots, v_k), \vec{w} = (w_1, \ldots, w_k) \in \mathcal{V}_k$. Here we calculate indices modulo k. Indeed, $\approx$ is an equivalence relation since $\vec{v} \approx \vec{w}$ implies that for all i the edge sequence of v_i and the edge sequence of w_i coincide. From this fact, symmetry and transitivity follow immediately.

We can interpret the $\mathcal{G}$-factorization $v = x_1 g_1 \cdots x_n g_n$ as k-tuple $\vec{v} \in \mathcal{V}_k$ with $k = n$ by setting $v_i = x_i g_i$ for all i – and likewise $\vec{w}$. We have $|\vec{v}|, |\vec{w}| \leq N$. Moreover, if $v \sim_{F(\mathcal{G})} w$, then there is a cyclic permutation $\vec{v}'$ of $\vec{v}$ with $\vec{v}' \approx \vec{w}$ (note that the converse implication is not true, in general).

The algorithm for solving the conjugacy problem consists of two phases. The first phase starts with the above described $\vec{v}$ and $\vec{w}$ and modifies them successively. It uses up to $\log n$ iterations in order to decide whether there is a cyclic permutation $\vec{v}'$ of $\vec{v}$ with $\vec{v}' \approx \vec{w}$. We show that this first phase can be done within time $\mathcal{O}(|\mathcal{A}|^3 \cdot \log n \cdot T(4N))$.

The following invariants are kept during the execution of the whole algorithm:

(I) $\vec{w} =_{F(\mathcal{G})} w$ and $\vec{v} =_{F(\mathcal{G})} v$ (actually, we replace v by cyclic permutations during the execution of the algorithm),

(II) if $v \sim_{F(\mathcal{G})} w$, then there is a cyclic permutation $\vec{v}'$ of $\vec{v}$ with $\vec{v}' \approx \vec{w}$.

As already remarked, these invariants hold at the beginning.

Pseudocode for the first phase of the algorithm can be found in Algorithm 6.2. It starts by determining some ℓ with $|v_\ell| \leq |v_i|$ for all i (in linear time). Next, two vectors $\alpha, \beta \in \{0,1\}^k$ are computed where

$$\alpha_i = \begin{cases} 1 & \text{if there are } a_i \in G_i^{\vec{v}}, b_i \in G_{i+1}^{\vec{v}} \text{ with } v_i =_{F(\mathcal{G})} a_i v_\ell b_i \\ 0 & \text{otherwise,} \end{cases}$$

$$\beta_i = \begin{cases} 1 & \text{if there are } a_i' \in G_i^{\vec{w}}, b_i' \in G_{i+1}^{\vec{w}} \text{ with } w_i =_{F(\mathcal{G})} a_i' v_\ell b_i' \\ 0 & \text{otherwise.} \end{cases}$$

By Remark 6.9, each check whether $v_i =_{F(\mathcal{G})} a_i v_\ell b_i$ for fixed $a_i \in G_i^{\vec{v}}, b_i \in G_{i+1}^{\vec{v}}$ can be performed in time $C \cdot |\mathcal{A}| \cdot T(|v_i| + |v_\ell| + 2)$ for some constant C. Hence, computing α requires at most

$$\begin{aligned} |\mathcal{A}|^2 \cdot \sum_{i=1}^{k} C \cdot |\mathcal{A}| \cdot T(|v_i| + |v_\ell| + 2) &\leq C \cdot |\mathcal{A}|^3 \cdot \sum_{i=1}^{k} T\left(|v_i| + \frac{N}{k} + 2\right) \\ &\leq C \cdot |\mathcal{A}|^3 \cdot T(4N) \end{aligned} \tag{6.1}$$

time for $N \geq 1$, and likewise for β. Thus, it takes at most $\mathcal{O}(|\mathcal{A}|^3 \cdot T(4N))$ time to compute both vectors.

Algorithm 6.2 *PhaseOne*

function PhaseOne($v = x_1 g_1 \cdots x_n g_n$, $w = y_1 h_1 \cdots y_n h_n \in \Pi(\mathcal{G})$)
begin
  $\vec{v} \leftarrow (x_1 g_1, \ldots, x_n g_n)$; $\vec{w} \leftarrow (y_1 h_1, \ldots, y_n h_n)$
  $k \leftarrow n$
  while (true) **do**
    choose ℓ such that $|v_\ell| \leq |v_i|$ for all i
    for $i \leftarrow 1$ **to** k **do**
      $$\alpha_i \leftarrow \begin{cases} 1 & \text{if there are } a_i \in G_i^{\vec{v}}, b_i \in G_{i+1}^{\vec{v}} \text{ with } v_i =_{F(\mathcal{G})} a_i v_\ell b_i \\ 0 & \text{otherwise} \end{cases}$$
      $$\beta_i \leftarrow \begin{cases} 1 & \text{if there are } a_i' \in G_i^{\vec{w}}, b_i' \in G_{i+1}^{\vec{w}} \text{ with } w_i =_{F(\mathcal{G})} a_i' v_\ell b_i' \\ 0 & \text{otherwise} \end{cases}$$
    endfor
    $(i_1, \ldots, i_m) \leftarrow \mathrm{KMP}(\alpha\alpha, \beta)$ (∗ find all positions of factors β in $\alpha\alpha$ ∗)
    if ($(i_1, \ldots, i_m)$ is empty) **then return** "not conjugate"
    if ($\alpha_i = \beta_i = 1$ for all i) **then return** PhaseTwo($\vec{v}, \vec{w}, \ell$)
    $d \leftarrow i_2 - i_1$ (∗ we know that $d \geq 2$ ∗)
    $\vec{v} \leftarrow (v_{i_1} \cdots v_{i_1+d-1}, \ldots, v_{i_1-d} \cdots v_{i_1-1})$ (∗ replace v by a cyclic permutation ∗)
    $\vec{w} \leftarrow (w_1 \cdots w_d, \ldots, w_{k-d+1} \cdots w_k)$
    $k \leftarrow k/d$
  endwhile
end

The first phase of the algorithm to decide conjugacy. Here, $\mathrm{KMP}(\alpha\alpha, \beta)$ returns a list of all indices in ascending order where a factor β starts in $\alpha\alpha$.

If $\alpha_i = \beta_i = 1$ for all i, the first phase is finished and the second phase starts (see Algorithm 6.3 below). Otherwise, there is some i with $\alpha_i = 0$ or $\beta_i = 0$. Now, the Knuth-Morris-Pratt algorithm is applied in order to find all cyclic permutations α' of α such that $\alpha' = \beta$ (this is done by searching the pattern β in the text $\alpha\alpha$ – taking time $\mathcal{O}(k) \subseteq \mathcal{O}(n)$). If there is no such cyclic permutation, we are done, because this implies that there is no cyclic permutation $\vec{v}'$ of $\vec{v}$ with $\vec{v}' \approx \vec{w}$; hence, v and w are not conjugate by the invariant (II).

Now, let $d > 0$ be minimal such that there is some i with $(\alpha_i, \ldots, \alpha_k, \alpha_1, \ldots, \alpha_{i-1}) = (\alpha_{i+d}, \ldots, \alpha_k, \alpha_1, \ldots, \alpha_{i+d-1}) = \beta$. We fix this i. It is straightforward to see that $d > 1$, and d divides k, and $\alpha_j = \alpha_{j+d}$ for all j. In particular, the "start positions" of β in $\alpha\alpha$ computed by the Knuth-Morris-Pratt algorithm are $(i + d\mathbb{Z}) \cap \{1, \ldots, k+1\}$. Moreover, if there is a cyclic permutation $\vec{v}'$ of $\vec{v}$ with $\vec{v}' \approx \vec{w}$, then $\vec{v}' = (v_{i+jd}, \ldots, v_k, v_1, \ldots, v_{i+jd-1})$ for some $j \in \mathbb{Z}$. This is because if $\vec{v}' \approx \vec{w}$, then the underlying edge sequences of v and w are the same; in particular, $G_i^{\vec{v}'} = G_i^{\vec{w}}$ for all i. Thus, $\vec{v}' \approx \vec{w}$ implies that β coincides

with the respective cyclic permutation of α. We set

$$\vec{v}_{\text{new}} = (v_i \cdots v_{i+d-1},\ \ldots\ , v_{i-d} \cdots v_{i-1}),$$
$$\vec{w}_{\text{new}} = (w_1 \cdots w_d,\ \ldots\ , w_{k-d+1} \cdots w_k),$$

where indices are calculated modulo k. We also replace v by the respective cyclic permutation; hence, we have again $\vec{v}_{\text{new}} =_{F(\mathcal{G})} v$ and $\vec{w}_{\text{new}} =_{F(\mathcal{G})} w$. Note that $|\vec{v}_{\text{new}}| = |\vec{v}|$ and $|\vec{w}_{\text{new}}| = |\vec{w}|$. Furthermore, $\vec{v}_{\text{new}}, \vec{w}_{\text{new}} \in \mathcal{V}_{k_{\text{new}}}$ for $k_{\text{new}} = k/d \leq k/2$, and we have the implications:

$$\begin{aligned} v \sim_{F(\mathcal{G})} w &\implies \exists\, \text{cyclic permutation } \vec{v}' \text{ of } \vec{v} \text{ with } \vec{v}' \approx \vec{w} \\ &\implies \exists\, \text{cyclic permutation } \vec{v}'_{\text{new}} \text{ of } \vec{v}_{\text{new}} \text{ with } \vec{v}'_{\text{new}} \approx \vec{w}_{\text{new}}. \end{aligned}$$

Now, we can treat $\vec{v}_{\text{new}}$ and $\vec{w}_{\text{new}}$ recursively. Hence, we have a total running time for the first phase in $\mathcal{O}(|\mathcal{A}|^3 \cdot \log n \cdot T(4N))$.

In the case $\alpha_i = \beta_i = 1$ for all i, we continue with the second phase, Algorithm 6.3. Recall that ℓ is an index such that $|v_\ell| \leq |v_i|$ for all i. As $\alpha_i = \beta_i = 1$ for all i, we know that there are $a_i \in G_i^{\vec{v}}, b_i \in G_{i+1}^{\vec{v}}, a'_i \in G_i^{\vec{w}}, b'_i \in G_{i+1}^{\vec{w}}$ with $v_i =_{F(\mathcal{G})} a_i v_\ell b_i$ and $w_i =_{F(\mathcal{G})} a'_i v_\ell b'_i$. In particular, the edge sequences of v_ℓ, v_i, and w_i coincide for all i. Hence, $G_i^{\vec{v}} = G_\ell^{\vec{v}}$ and likewise $G_i^{\vec{w}} = G_\ell^{\vec{v}}$ for all i. Thus, we have $\vec{v} \approx \vec{w}$. We set $A = G_\ell^{\vec{v}}$.

Let $\prec$ be some arbitrary total order on A. We will use the lexicographic order on A^k, i. e., $(a_1, \ldots, a_k) \prec (a'_1, \ldots, a'_k)$ if and only if there is some $1 \leq i \leq k$ such that $a_j = a'_j$ for $j < i$ and $a_i \prec a'_i$.

The first step of the second phase is to compute the lexicographically smallest vector $(\hat{a}_1, \ldots, \hat{a}_k) \in A^k$ such that there is some $\hat{b}_k \in A$ with

$$v =_{F(\mathcal{G})} \hat{a}_1 v_\ell \cdots \hat{a}_k v_\ell \cdot \hat{b}_k.$$

Such a vector exists because we have $v =_{F(\mathcal{G})} a_1 v_\ell \cdot [b_1 a_2] v_\ell \cdots [b_{k-1} a_k] v_\ell \cdot b_k$. Moreover, it can be computed by checking from left to right for every $\hat{a}_i \in A$ whether there is some $\hat{b}_i$ with $\hat{b}_{i-1} v_i =_{F(\mathcal{G})} \hat{a}_i v_\ell \hat{b}_i$ (where $\hat{b}_0 = 1$) and then choosing the $\prec$-smallest such $\hat{a}_i$. As in (6.1), this can be done in time $\mathcal{O}(|\mathcal{A}|^3 \cdot T(4N))$ using Remark 6.9.

Thus, for the following, we may assume that $v = \hat{a}_1 v_\ell \cdots \hat{a}_k v_\ell \cdot \hat{b}_k$ and $\vec{v} = (\hat{a}_1 v_\ell, \ldots, \hat{a}_{k-1} v_\ell, \hat{a}_k v_\ell \hat{b}_k)$. Next, we proceed as in [DDM12, Prop. 6.1] and extend this lexicographic "normal form" to vv: We compute the lexicographically smallest vector $\zeta = (\hat{a}_1, \ldots, \hat{a}_{2k-1}) \in A^{2k-1}$ (i. e., it remains to compute $\hat{a}_{k+1}, \ldots, \hat{a}_{2k-1}$, only) such that there are $\hat{a}_{2k}, \hat{b}_{2k} \in A$ with

$$vv =_{F(\mathcal{G})} \hat{a}_1 v_\ell \cdots \hat{a}_{2k-1} v_\ell \cdot \hat{a}_{2k} v_\ell \cdot \hat{b}_{2k}.$$

Again, it can be computed in time $\mathcal{O}(|\mathcal{A}|^3 \cdot T(4N))$. For $k < i < 2k$, we define $\hat{b}_i \in A$ such that

$$\underbrace{\hat{a}_1 v_\ell \cdots \hat{a}_k v_\ell \cdot \hat{b}_k}_{=v} \cdot \hat{a}_1 v_\ell \cdots \hat{a}_{i-k} v_\ell =_{F(\mathcal{G})} \hat{a}_1 v_\ell \cdots \hat{a}_i v_\ell \cdot \hat{b}_i.$$

Algorithm 6.3 *PhaseTwo*

function PhaseTwo($\vec{v}, \vec{w} \in \mathcal{V}^k$, $\ell \in \mathbb{N}$)
begin
  $A \leftarrow G_\ell^{\vec{v}}$; $\hat{b}_0 \leftarrow 1 \in A$
  for $i \leftarrow 1$ **to** k **do**
    $\hat{a}_i \leftarrow \min\left\{a \in A \;\middle|\; \exists\, b \in A : av_\ell b = \hat{b}_{i-1}v_i\right\}$; $\hat{b}_i \leftarrow$ such that $\hat{a}_i v_\ell \hat{b}_i = \hat{b}_{i-1}v_i$
  endfor
  for $i \leftarrow k+1$ **to** $2k-1$ **do**
    $\hat{a}_i \leftarrow \min\left\{a \in A \;\middle|\; \exists\, b \in A : av_\ell b = \hat{b}_{i-1}\hat{a}_{i-k}v_\ell\right\}$; $\hat{b}_i \leftarrow$ such that $\hat{a}_i v_\ell \hat{b}_i = \hat{b}_{i-1}\hat{a}_{i-k}v_\ell$
  endfor
  $\zeta \leftarrow (\hat{a}_1, \ldots, \hat{a}_{2k-1})$
  for each $a \in A$ **do**
    $b_0^{(a)} \leftarrow a$
    for $i \leftarrow 1$ **to** k **do**
      $a_i^{(a)} \leftarrow \min\left\{a \in A \;\middle|\; \exists\, b \in A : av_\ell b = b_{i-1}^{(a)}w_i\right\}$; $b_i^{(a)} \leftarrow$ such that $a_i^{(a)} v_\ell b_i^{(a)} = b_{i-1}^{(a)} w_i$
    endfor
    $\xi^{(a)} \leftarrow (a_1^{(a)}, \ldots, a_k^{(a)})$; $b_k^{(a)} \leftarrow b_k^{(a)} \cdot a^{-1}$
    $(i_1, \ldots, i_m) \leftarrow \mathrm{KMP}(\zeta, \xi^{(a)})$ (∗ find all positions of factors $\xi^{(a)}$ in ζ ∗)
    if (there is some j with $b_k^{(a)} = \hat{b}_{i_j+k-1}$) **then return** "conjugate"
  endfor
  return "not conjugate"
end

The second phase of the algorithm to decide conjugacy. Here, the minimum $\min\{a \in A \mid \exists\, b \in A : av_\ell b = \hat{b}_{i-1}v_i\}$ is with respect to the order $\prec$ on A.

These $\hat{b}_i$ are computed as a by-product of the normal form computation.

Now, using the same method we compute for every $a \in A$ the lexicographically smallest $\xi^{(a)} = \left(a_1^{(a)}, \ldots, a_k^{(a)}\right) \in A^k$ such that there is some $b_k^{(a)} \in A$ with

$$awa^{-1} =_{F(\mathcal{G})} a_1^{(a)} v_\ell \cdots a_k^{(a)} v_\ell \cdot b_k^{(a)}.$$

The final step is to apply the Knuth-Morris-Pratt algorithm over the alphabet A to find out whether $\xi^{(a)}$ matches with some substring of ζ (again taking time $\mathcal{O}(k) \subseteq \mathcal{O}(n)$). For every matching

$$\left(a_1^{(a)}, \ldots, a_k^{(a)}\right) = (\hat{a}_i, \ldots, \hat{a}_{i+k-1})$$

with $1 \leq i < k$, we have to check whether $b_k^{(a)} = \hat{b}_{i+k-1}$. If this is the case, we have

$$\begin{aligned} awa^{-1} &=_{F(\mathcal{G})} a_1^{(a)} v_\ell \cdots a_k^{(a)} v_\ell \cdot b_k^{(a)} \\ &=_{F(\mathcal{G})} \hat{a}_i v_\ell \cdots \hat{a}_{i+k-1} v_\ell \cdot \hat{b}_{i+k-1} \end{aligned}$$

$$\begin{aligned}&=_{F(\mathcal{G})} \hat{a}_i v_\ell \cdots \hat{a}_k v_\ell \cdot \hat{b}_k \cdot \hat{a}_1 v_\ell \cdots \hat{a}_{i-1} v_\ell \\ &=_{F(\mathcal{G})} v'\end{aligned}$$

for some cyclic permutation v' of v; hence, $v \sim_{F(\mathcal{G})} w$. On the other hand, if $awa^{-1} =_{F(\mathcal{G})} v'$ for some cyclic permutation v' of v, then, by the invariant (II), there is some i with $awa^{-1} =_{F(\mathcal{G})} v' = \hat{a}_i v_\ell \cdots \hat{a}_k v_\ell \cdot \hat{b}_k \cdot \hat{a}_1 v_\ell \cdots \hat{a}_{i-1} v_\ell$; hence,

$$\begin{aligned}a_1^{(a)} v_\ell \cdots a_k^{(a)} v_\ell \cdot b_k^{(a)} &=_{F(\mathcal{G})} awa^{-1} =_{F(\mathcal{G})} v' \\ &= \hat{a}_i v_\ell \cdots \hat{a}_k v_\ell \cdot \hat{b}_k \cdot \hat{a}_1 v_\ell \cdots \hat{a}_{i-1} v_\ell \\ &=_{F(\mathcal{G})} \hat{a}_i v_\ell \cdots \hat{a}_{i+k} v_\ell \cdot \hat{b}_{i+k-1}.\end{aligned}$$

Since both, the first and the last line, are lexicographically smallest, it follows that there is a matching $\left(a_1^{(a)}, \ldots, a_k^{(a)}\right) = (\hat{a}_i, \ldots, \hat{a}_{i+k-1})$ and $b_k^{(a)} = \hat{b}_{i+k-1}$. □

Now, we can combine Corollary 6.10 and Theorem 6.11 and we obtain:

Theorem 6.12. *Let $\mathcal{G}$ be a finite graph of groups with finite edge groups. Assume that the conjugacy problem in the vertex groups G_P for all $P \in V(Y)$ is decidable in time $T(N)$. Then, there is an algorithm taking as input two $\mathcal{G}$-factorizations $v, w \in \Sigma^*$ with $|v|, |w| \leq N$ and $|v|_{\mathcal{G}}, |w|_{\mathcal{G}} \leq n$, which decides whether v and w are conjugate in time $\mathcal{O}(|\mathcal{A}|^3 \cdot \log n \cdot T(4N))$.*

Corollary 6.13. *Let $\mathcal{G}$ be a finite graph of groups with finite edge groups. If the conjugacy problem in G_P for all $P \in V(Y)$ is decidable in time $\mathcal{O}(N \cdot \log^k N)$ for some $k \in \mathbb{N}$, then the conjugacy problem of $\pi_1(\mathcal{G})$ is decidable in time $\mathcal{O}(N \cdot \log^{k+1} N)$.*

Until now, we only considered a fixed finite graph of groups. A natural question is how the complexity changes if also the graph of groups is part of the input. As a preparation for the answer of this question, we already analyzed the running time also in dependence of the total size of the edge groups $|\mathcal{A}|$. As we have seen, the running time for the word and conjugacy problem in $\pi_1(\mathcal{G})$ does only depend on $|\mathcal{A}|$ and the bound for the running time in the vertex groups $T(N)$. However, we assumed $T(N)$ to be a bound for the running time for *all* vertex groups. If we allow the vertex groups to be from an infinite set of groups, it would be very restricting to assume that in all these groups the running time for deciding the conjugacy problem is bounded by $T(N)$. Therefore, we let T depend on an additional parameter M where M is the size of a the presentation of the respective vertex group. We assume that T is monotone in M.

For case (i) of Theorem 3.19, we now need to determine in addition which elements of $\mathcal{A}$ are conjugate. This requires at most $|\mathcal{A}|^2$ additional calls to the conjugacy problem of the vertex groups, each taking time at most $T(M, 2)$. Hence, we conclude this section with the following theorem.

Theorem 6.14. *Let $\mathcal{D}$ be a class of groups such that there is an algorithm which decides conjugacy in time $T(M, N)$ on input of a group $G' \in \mathcal{D}$ and two input words of length N where G' is given via a presentation $G' = \Sigma'^*/S'$ with $|\Sigma'| + |S'| \leq M$.*
Then, there is an algorithm which decides the conjugacy problem of $\pi_1(\mathcal{G})$ in time $\mathcal{O}(|\mathcal{A}|^3 \cdot \log n \cdot T(M, 4N))$ if the input consists of a finite graph of groups given by

- *a finite graph Y in a proper encoding,*
- *for every $P \in V(Y)$, a finite symmetric presentation $G_P = \Sigma_P^*/S_P$ of the vertex group $G_p \in \mathcal{D}$ with $|\Sigma_P| + |S_P| \leq M$ such that $G_y^y \subseteq \Sigma_{\iota(y)}$ for all $y \in E(Y)$,*
- *for every $y \in E(Y)$, a table storing the homomorphism $G_y \to G_y^y$,*

and two $\mathcal{G}$-factorizations $v, w \in \Sigma^*$ with $|v|, |w| \leq N$ and $|v|_{\mathcal{G}}, |w|_{\mathcal{G}} \leq n$.

Again, we remark that we use the bound $|\mathcal{A}|^3$ in Theorem 6.14 only for simplicity. It is only tight if the graph Y consists of only one vertex and one edge.

6.2. Circuit Complexity for Finite Edge Groups

In this section, we want to consider the circuit complexity of the word and conjugacy problem of fundamental groups of finite graphs of groups with finite edge groups. We generalize the result by Waack that the word problem of a free product is NC^1-Turing-reducible to the word problem of the factors and the word problem of the free group F_2 ([Waa90]).

Let $\mathcal{L}$ be a finite set of languages. Recall that $L \in \mathsf{NC}^2(\mathcal{L})$ if and only if there is some NC^2-circuit using also oracle gates for languages in $\mathcal{L}$, where each oracle gate with k inputs counts as depth $\log k$.

Theorem 6.15. *Let $\mathcal{G}$ be a finite graph of groups with finite edge groups (with underlying graph Y). Then*

$$\mathrm{WP}(\pi_1(\mathcal{G})) \in \mathsf{NC}^2(\{\mathrm{WP}(G_P) \mid P \in V(Y)\}) \qquad \text{and}$$
$$\mathrm{CP}(\pi_1(\mathcal{G})) \in \mathsf{NC}^2(\{\mathrm{CP}(G_P) \mid P \in V(Y)\})\,.$$

Corollary 6.16. *Let $\mathcal{G}$ be a finite graph of groups with finite edge groups. If the word (resp. conjugacy) problem of every vertex group G_P for $P \in V(Y)$ is in NC. Then the word (resp. conjugacy) problem in $\pi_1(\mathcal{G})$ is in NC.*

Proof of Theorem 6.15. We first consider the word problem of $\pi_1(\mathcal{G}, P)$ for some $P \in V(Y)$. The conjugacy problem then follows rather easily. We construct an NC^2 circuit which also uses gates for the word problem of the vertex groups (but only $\log n$ such gates on any path from the input gates to the output gates). The circuit for a word of $\mathcal{G}$-length n consists of $\log n$ stages. Each of them takes as input two Britton-reduced $\mathcal{G}$-factorizations u, v and outputs a Britton-reduced $\mathcal{G}$-factorization $\widehat{uv}$ with $\widehat{uv} =_{F(\mathcal{G})} uv$. Hence, for our circuit, the task in each stage is to find a suffix z of u which is maximal with the property that there is a prefix $\tilde{z}$ of v and some $a \in \mathcal{A}$ with $z^{-1}a =_{F(\mathcal{G})} \tilde{z}$.

A single stage works as follows: Let $u = g_0 x_1 g_1 \cdots x_m g_m \in \Pi(\mathcal{G}, Q, R)$ and $v = h_0 y_1 h_1 \cdots y_n h_n \in \Pi(\mathcal{G}, R, S)$ be the two Britton-reduced input words for some $Q, R, S \in V(Y)$. For $0 \leq i \leq j < \min\{m, n\}$ and $a, b \in \mathcal{A}$, we define

$$B(i, j; a; b) = \begin{cases} \text{true} & \text{if } x_{m-j} g_{m-j} \cdots x_{m-i} g_{m-i} \cdot a \cdot h_i y_{i+1} \cdots h_j y_{j+1} =_{F(\mathcal{G})} b, \\ \text{false} & \text{otherwise.} \end{cases}$$

Now, our aim is to compute the maximal j such that there is some $b \in \mathcal{A}$ with $B(0, j; 1; b) = \text{true}$. The truth values $B(i, j; a; b)$ can be computed recursively via

$$B(i, i; a; b) = \begin{cases} \text{true} & \text{if } x_{m-i} g_{m-i} \cdot a \cdot h_i y_{i+1} =_{F(\mathcal{G})} b, \\ \text{false} & \text{otherwise,} \end{cases}$$

and, for $i < j$,

$$B(i, j; a; b) = \bigvee_{c \in \mathcal{A}} \left(B\left(i, \left\lfloor \frac{i+j}{2} \right\rfloor; a; c\right) \wedge B\left(\left\lfloor \frac{i+j}{2} \right\rfloor + 1, j; c; b\right)\right).$$

The only place where oracle gates for the word problem in the vertex groups are used is for $B(i, i; a; b)$. As $|\mathcal{A}|$ is a constant, every stage consists of several calls to the word problem of the vertex groups – all in parallel – followed by a NC^1 circuit. Hence, every stage is a $\mathsf{NC}^1(\{\mathrm{WP}(G_P) \mid P \in V(Y)\})$ circuit. As there are $\log n$ stages, the result for the word problem follows. Moreover, this circuit computes a Britton-reduced word; hence, by Lemma 3.18, it can be used to reduce $\mathcal{G}$-factorizations cyclically.

For the conjugacy problem, consider again the three cases of the Conjugacy Criterion, Theorem 3.19. We assume that the inputs are already cyclically reduced. In case (i) and (ii), the circuits for the conjugacy problem of the vertex groups can be used (at most $2|\mathcal{A}| + 1$ instances in the vertex groups have to be solved). In case (iii), all cyclic permutations and also all conjugations with some $a \in \mathcal{A}$ can be handled in parallel; hence, it reduces to solving another instance of the word problem in $\pi_1(\mathcal{G}, P)$ for some $P \in V(Y)$ (indeed, the last stage of the circuit for the word problem is sufficient as both words are already cyclically reduced). □

Note that Theorem 6.15 implies that virtually free groups have conjugacy problem in NC^2. However, this is not optimal. In fact, free groups have word problem in $\mathsf{LOGSPACE}$. Theorem 5.2 of [Rob93] implies that also virtually free groups have word and conjugacy problem in $\mathsf{LOGSPACE}$ (see also [EEO13]). This leads to the question whether NC^2 can be replaced by NC^1 in Theorem 6.15. In the case of free products, this is actually true as Waack proved in [Waa90] (see also [DK14]).

Theorem 6.17 ([Waa90])**.** *Let $\mathcal{G}$ be a finite graph of groups with trivial edge groups (i. e., $\pi_1(\mathcal{G})$ is a free product of the vertex groups and some infinite cyclic groups). Then*

$$\mathrm{WP}(\pi_1(\mathcal{G})) \in \mathsf{NC}^1(\{\mathrm{WP}(G_P) \mid P \in V(Y)\} \cup \{\mathrm{WP}(F_2)\}),$$

where F_2 denotes the free group with two generators.

We can extend this to some more cases similarly as in [EEO13, Cor. 22] where word problems in $\mathsf{LOGSPACE}$ are considered.

Corollary 6.18. *Let $\mathcal{G}$ be a finite graph of groups with finite edge groups. Moreover, assume that one of the following two conditions is satisfied:*

(i) *For all $P \in V(Y)$, there is some finite normal subgroup $H_P \leq G_P$ such that $H_P =_{G_P} G_y^y$ for all $y \in E(Y)$ with $\iota(y) = P$.*

(ii) *For all $P \in V(Y)$, there is some finite group H_P and a homomorphism $\varphi_P : G_P \to H_P$ such that the restriction of φ_P to G_y^y is injective for all $y \in E(Y)$ with $\iota(y) = P$.*

Then we have

$$\mathrm{WP}(\pi_1(\mathcal{G})) \in \mathsf{NC}^1(\{\mathrm{WP}(G_P) \mid P \in V(Y)\} \cup \{\mathrm{WP}(F_2)\}) \qquad \textit{and}$$
$$\mathrm{CP}(\pi_1(\mathcal{G})) \in \mathsf{NC}^1(\{\mathrm{CP}(G_P) \mid P \in V(Y)\} \cup \{\mathrm{WP}(F_2)\})\,.$$

In particular, if the word resp. conjugacy problem for all vertex groups G_P for $P \in V(Y)$ is in LOGSPACE *(resp.* NC^{i+1} *or* AC^i *for some $i \geq 1$). Then the word resp. conjugacy problem in $\pi_1(\mathcal{G})$ is in* LOGSPACE *(resp.* NC^{i+1} *or* AC^i*).*

Proof. The conjugacy problem follows from the word problem as in the proof of Theorem 6.15. Hence, we only need to consider the word problem.

(i): Let $H = H_P$ for some $P \in V(Y)$. We consider the fundamental group $\pi_1(\mathcal{G}, T)$ with respect to some spanning tree T. Then $H = H_Q$ as a subgroup of $\pi_1(\mathcal{G}, T)$ for all $Q \in V(Y)$. This implies that H is normal in $\pi_1(\mathcal{G}, T)$. Now, $\pi_1(\mathcal{G}, T)/H = \pi_1(\mathcal{G}', T)$ for some graph of groups $\mathcal{G}'$ with the same underlying graph Y but with trivial edge groups. By Theorem 6.17, the word problem of $\pi_1(\mathcal{G}, T)/H$ is in NC^1 $(\{\mathrm{WP}(G_P), \mathrm{WP}(F_2) \mid P \in V(Y)\})$. If for some word w we have $w \neq_{\pi_1(\mathcal{G},T)/H} 1$, then, in particular, $w \neq_{\pi_1(\mathcal{G},T)} 1$. Hence, we may assume $w \in H$. As H is finite, the word problem of H is in NC^1. This concludes the first case.

(ii): Let $X = \prod_{P \in V(Y)} H_P$ be the cartesian product of the H_Ps. Then every vertex group G_P acts on X such that for all $y \in E(Y)$ the induced action of G_y^y on X is free. By Lemma II.2.5 and Proposition II.2.2 of [Dic80], these actions induce a homomorphism $\varphi : \pi_1(\mathcal{G}) \to \mathrm{Sym}(X)$ (where $\mathrm{Sym}(X)$ denotes the group of bijections $X \to X$). Moreover, the restriction for φ to G_y^y is injective for all $y \in E(Y)$.

The kernel N of φ has trivial intersection with all edge groups, and it acts with finitely many orbits on the Bass-Serre tree of $\pi_1(\mathcal{G})$. Hence, by Theorem 3.12, the kernel N satisfies the requirements of Theorem 6.17. In [Rob93, Thm. 5.2], Robinson showed that the word problem of some group is NC^1-reducible to the word problem of a finite index subgroup. Hence, we are done. □

Chapter 7.

Conjugacy in Baumslag-Solitar Groups

A *Baumslag-Solitar group* is a group of the form $\mathbf{BS}_{p,q} = \langle\, a, t \mid ta^pt^{-1} = a^q \,\rangle$ for $p, q \in \mathbb{Z} \setminus \{0\}$. W. l. o. g. we assume that $1 \leq p \leq |q|$. These groups were introduced in 1962 by Baumslag and Solitar [BS62] as examples for finitely presented non-Hopfian two-generator groups. They showed that the class of Baumslag-Solitar groups comprises both Hopfian and non-Hopfian groups. In our case, we are interested in another distinction between different Baumslag-Solitar groups based on the following well-known fact:

Remark 7.1. A Baumslag-Solitar group $\mathbf{BS}_{p,q}$ with $1 \leq p \leq |q|$ is solvable if and only if $p = 1$.

Solvable Baumslag-Solitar groups have a very simple structure; indeed, they are metabelian. The next section is dedicated to the solution of their conjugacy problem, which turns out to be in TC^0. After that, we proceed to a more general setting and examine the complexity of the word problem (in Section 7.3) and conjugacy problem (in Section 7.4) of generalized Baumslag-Solitar groups. The main obstacle for solving the conjugacy problem in $\mathsf{LOGSPACE}$ is the word problem: We can only show that it is in $\mathsf{LOGDCFL}$.

7.1. Conjugacy in Solvable Baumslag-Solitar Groups

Robinson showed in [Rob93] that the word problem of $\mathbf{BS}_{1,q}$ is TC^0-complete (w. r. t. AC^0 reductions). Here, we prove that also the conjugacy problem of $\mathbf{BS}_{1,q}$ is TC^0-complete. The results for the solution of the conjugacy problem in the Baumslag-Solitar group $\mathbf{BS}_{1,2}$ will also be used in Chapter 8 when dealing with the Baumslag group $\mathbf{G}_{1,2}$.

In $\mathbf{BS}_{1,q}$, we have the equalities $ta = a^qt$ and $at^{-1} = t^{-1}a^q$. This allows to represent all group elements by words of the form $t^{-k}a^\ell t^m$ with $k, m \in \mathbb{N}$ and $\ell \in \mathbb{Z}$. However, for $m \geq 0$, transforming $t^m a^\ell$ into this form, leads to $a^r t^m$ with $r = q^m\ell$; so the word $a^r t^m$ can be exponentially longer than the word $t^m a^\ell$.

We denote by $\mathbb{Z}[1/q] = \{\ell/q^m \in \mathbb{Q} \mid \ell, m \in \mathbb{Z}\}$ the set of fractions having a power of q as denominator (in the case $q = 2$, $\mathbb{Z}[1/q]$ is called the ring of *dyadic fractions*). Multiplication by q is an automorphism of the underlying additive group, and therefore, we can define the semidirect product $\mathbb{Z}[1/q] \rtimes \mathbb{Z}$ as follows: Elements are pairs $(r, m) \in \mathbb{Z}[1/q] \times \mathbb{Z}$; the multiplication in $\mathbb{Z}[1/q] \rtimes \mathbb{Z}$ is defined by

$$(r, m) \cdot (s, k) = (r + q^m s, m + k).$$

Inverses can be computed by the formula $(r,m)^{-1} = (-r \cdot q^{-m}, -m)$. It is straightforward to see that $a \mapsto (1,0)$ and $t \mapsto (0,1)$ defines an isomorphism $\mathbf{BS}_{1,q} \to \mathbb{Z}[1/2] \rtimes \mathbb{Z}$.

There are several options to represent a group element $g \in \mathbf{BS}_{1,q}$. As usual when dealing with word and conjugacy problems, group elements can be written as words over some alphabet with involution. We call this the *unary representation* of group elements. For $\mathbf{BS}_{1,q}$, we use the alphabet $\{a, \bar{a}, t, \bar{t}\}$. Another way is to use the decomposition as a semidirect product and to write $g = (r,m)$ with $r \in \mathbb{Z}[1/q]$ and $m \in \mathbb{Z}$. The *binary representation* of (r,m) consists of r written as (binary) floating point number with the exponent being to base q and m in unary. Let us write (r,m) with $r = q^k s$ and $k, s, m \in \mathbb{Z}$. Then we have $(q^k s, m) = (0,k) \cdot (s, m-k)$ and the corresponding triple $[k, s, m-k] \in \mathbb{Z}^3$ is called the *triple representation* of (r,m); it is not unique. The triple representation will be important when solving the conjugacy problem in the Baumslag group, see Chapter 8.

Note that if $w \in \{a, \bar{a}, t, \bar{t}\}^N$ satisfies $w =_{\mathbf{BS}_{1,q}} (r,m)$, then $|r| \leq q^N$ and $|m| \leq N$. Thus, a transformation from unary to binary notation is on the safe side as it leads only to a linear increase of size.

Proposition 7.2. *There is a uniform family of TC^0 circuits for the following problem: Given $(r_1,m_1), \ldots, (r_n,m_n) \in \mathbb{Z}[1/q] \rtimes \mathbb{Z}$ in binary representation for all i, calculate (r,m) with $(r,m) = (r_1,m_1) \cdots (r_n,m_n)$ in $\mathbb{Z}[1/q] \rtimes \mathbb{Z}$.*

Proof. The statements concerning computations in TC^0 are summarized in Theorem 2.4 (see also the textbook [Vol99]). We have $m = \sum_{i=1}^n m_i$. Using the equality $(x,\ell)(s,k) = (x + s \cdot q^\ell, \ell + k)$, we see that

$$r = \sum_{i=1}^{n} r_i \cdot q^{k_i} \qquad \text{for} \qquad k_i = \sum_{j=1}^{i-1} m_j.$$

The numbers k_i can be calculated by Iterated Addition of the unary numbers m_j for $j < i$, which is in TC^0. In particular, m can be calculated by a TC^0 circuit. The mappings $r_i \mapsto r_i \cdot q^{k_i}$ can be computed by TC^0 circuits, since the k_i are bounded by the length of the input and Iterated Multiplication is in TC^0 (Theorem 2.4) – in fact, this is a somehow easier task than Iterated Multiplication. It remains to calculate the Iterated Addition of binary numbers $r_i \cdot q^{k_i}$, which is in TC^0, again. □

The next proof uses the deep result by Hesse that Integer Division is in uniform TC^0 [Hes01, HAB02], see Theorem 2.4.

Proposition 7.3. *There is a uniform family of TC^0 circuits for the following problem: Given $f = (r,m), g = (s,k) \in \mathbb{Z}[1/q] \rtimes \mathbb{Z}$ in binary representation, decide whether $f \sim_{\mathbf{BS}_{1,q}} g$.*

Proof. Let $(r,m) \sim_{\mathbf{BS}_{1,q}} (s,k)$, i. e., there are $\ell \in \mathbb{Z}$, $x \in \mathbb{Z}[1/q]$ with $(x,\ell)(r,m) = (s,k)(x,\ell)$. In particular, $(r,m) \sim_{\mathbf{BS}_{1,q}} (s,k)$ if and only if $m = k$ and there are $\ell \in \mathbb{Z}$, $x \in \mathbb{Z}[1/q]$ such that

$$s = r \cdot q^\ell - x \cdot (q^m - 1). \tag{7.1}$$

We have $(r,m) \sim_{\mathbf{BS}_{1,q}} (s,m)$ if and only if $(-r,-m) \sim_{\mathbf{BS}_{1,q}} (-s,-m)$ since $(-y,-m) \sim_{\mathbf{BS}_{1,q}} (-y \cdot q^{-m},-m) = (y,m)^{-1}$ for all $y \in \mathbb{Z}[1/q]$. Therefore, without restriction $m \in \mathbb{N}$. Furthermore, since a conjugation with t^ℓ maps (r,m) to $(q^\ell r,m)$, we may assume that $r,s \in \mathbb{Z}$ and $m \in \mathbb{N}$. For $m=0$, this means $(r,0) \sim_{\mathbf{BS}_{1,q}} (s,0)$ if and only if there is some $\ell \in \mathbb{Z}$ such that $s = r \cdot q^\ell$. This can be checked in TC^0 by trying all possible (linearly many) values for ℓ in parallel (using Iterated Multiplication to compute q^ℓ).

For $m > 0$, our assumption $r,s \in \mathbb{Z}$ and (7.1) imply that $x \in \mathbb{Z}[1/q] \cap \frac{1}{q^m-1} \cdot \mathbb{Z} = \mathbb{Z}$. As $q^m \equiv 1 \mod q^m - 1$, we can reformulate (7.1):

$$\begin{aligned}(r,m) &\sim_{\mathbf{BS}_{1,q}} (s,m) \\ &\iff \exists\, \ell \in \{0,\dots,m-1\} \text{ such that } r \cdot q^\ell - s \equiv 0 \mod (q^m-1).\end{aligned} \tag{7.2}$$

Since there are only linearly many possible values for ℓ, it can be checked in parallel whether such ℓ exists using Hesse's result for Integer Division, see Theorem 2.4. □

For the word problem and conjugacy problem, we usually assume the inputs to be given in unary. However, as we have seen, Proposition 7.2 and Proposition 7.3 also hold when the input is a (sequence of) binary representation(s).

Theorem 7.4. *The word problem as well as the conjugacy problem in $\mathbf{BS}_{1,q}$ is TC^0-complete when the input is given in unary or as sequence of elements given in binary.*

Note that we restrict the word problem to a single element in binary representation, the hardness part of Theorem 7.4 fails: in this case one only needs to check whether $(r,m) = (0,0)$.

Proof. By Proposition 7.2 and Proposition 7.3, the conjugacy problem can be solved in TC^0. The word problem of $\mathbb{Z}$ (input given in unary) is a special instance of the word problem in $\mathbf{BS}_{1,q}$ – also if the input is given as a sequence of binary representations. Now, the word problem in $\mathbb{Z}$ in unary notation is TC^0-hard because the TC^0-hard problem Majority (see Remark 2.3) reduces uniformly to the unary word problem in $\mathbb{Z}$. □

7.2. Generalized Baumslag-Solitar Groups

A *generalized Baumslag-Solitar group* (*GBS group*) is a fundamental group of a finite graph of groups with only infinite cyclic vertex and edge groups. That means a GBS group $G = \pi_1(\mathcal{G})$ is completely given by a finite graph Y and numbers $\alpha_t, \beta_t \in \mathbb{Z} \setminus \{0\}$ for $t \in E(Y)$ such that $\alpha_t = \beta_{\bar{t}}$. For $a \in V(Y)$ we write $G_a = \langle a \rangle$. Then we have

$$F(\mathcal{G}) = \left\langle V(Y), E(Y) \;\middle|\; t b^{\beta_t} \bar{t} = a^{\alpha_t}, \bar{t}t = 1 \text{ for } t \in E(Y), a = \iota(t), b = \tau(t) \right\rangle$$

and $G = \pi_1(\mathcal{G}, a) \leq F(\mathcal{G})$ for any $a \in V(Y)$ as in Definition 3.4. (Note that $V(Y) \cup E(Y)$ generates $F(\mathcal{G})$ as a group, but in general, *not* as a monoid.)

Baumslag-Solitar groups $\mathbf{BS}_{p,q}$ are the special case that Y consists of one vertex and one loop t with $\alpha_t = p$, $\beta_t = q$. In this case, we have $F(\mathcal{G}) = \mathbf{BS}_{p,q}$.

Lemma 7.5. *A GBS group G is solvable if and only if it is a solvable Baumslag-Solitar group* $\mathbf{BS}_{1,q}$.

Proof. Let $\mathcal{G}$ be a reduced graph of groups for G. As soon as $\mathcal{G}$ contains more than one edge, there is a free group of rank two inside G, and hence G is not solvable. The other direction was already shown in the preceding section. □

In [Wis00], Wise states without proof that GBS groups are *residually finite* if and only if they are solvable or if every cycle $t_1, \dots, t_n$ in the graph Y satisfies $\prod_{i=1}^{n} \frac{\beta_{t_i}}{\alpha_{t_i}} = \pm 1$. A proof can be found in [Lev14, Cor. 7.7]. For ordinary Baumslag-Solitar groups, this result was first proven by Meskin [Mes72]. Note that, also in [BS62], a result characterizing residually finite Baumslag-Solitar groups was stated. However, that characterization was not correct as Meskin showed in [Mes72].

Definition 7.6. A group G is called *linear* if it can be embedded into a matrix group over some field, i. e., if there is some field $\mathbb{F}$ and some $k \in \mathbb{N}$ and an injective homomorphism $G \to \mathrm{GL}(k, \mathbb{F})$ where GL is the general linear group.

Every linear group is residually finite [Mal40]. Moreover, for ordinary Baumslag-Solitar groups, it is well-known that they are linear if and only if they are residually finite.

It is also a well-known fact that virtually free groups are linear. To see this, take the embedding of the free group $F_{\{a,b\}}$ into $\mathrm{SL}(2, \mathbb{Z})$ via

$$a \mapsto \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, \qquad b \mapsto \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Now, linear groups are closed under finite extensions. In fact, let

$$1 \to H \to G \to Q \to 1$$

be an exact sequence with Q finite, and let $\varphi : H \to \mathrm{GL}(n, \mathbb{F})$ be some injective homomorphism for some field $\mathbb{F}$. In particular, H acts faithfully on $\mathbb{F}^n$. We construct the so-called *induced representation* for G. Let $q = |Q|$ and let $x_1, \dots, x_q \in G$ be a set of representatives of Q. We construct a faithful action of G on $\bigoplus_{i=1}^{q} \mathbb{F}^n x_i$. We can write every $g \in G$ in a unique way as $g = hx_i$ for some i and some $h \in H$. Moreover, for all i, j there is some $h_{i,j} \in H$ such that $x_i x_j = h_{i,j} x_{i,j}$ where $x_{i,j}$ denotes the representative of $x_i x_j \in Q$. For $v \in \mathbb{F}^n$ and $h \in H$, we set

$$\begin{aligned} x_i \cdot v x_j &= \varphi(h_{i,j}) v x_{i,j} \\ h \cdot v x_j &= \varphi(h) v x_j. \end{aligned}$$

This is a faithful action of G on $\mathbb{F}^{qn}$. Hence, it induces an injective homomorphism $G \to \mathrm{GL}(nq, \mathbb{F})$.

In [Lev07], Levitt showed that a GBS group is virtually $F \times \mathbb{Z}$ for some free group F if and only if every cycle $t_1, \dots, t_n$ in the graph Y satisfies

$$\prod_{i=1}^{n} \frac{\beta_{t_i}}{\alpha_{t_i}} = \pm 1.$$

Furthermore, in [Lev14, Cor. 7.7], he showed that for every GBS group G one of the following alternatives holds (which are mutually excluding unless $G = \mathbf{BS}_{1,1}$ or $G = \mathbf{BS}_{1,-1}$):

- G is a solvable Baumslag-Solitar group,
- G is virtually $F \times \mathbb{Z}$ for some free group F,
- G is non-residually finite.

As a direct consequence of these results, the fact that the class of linear groups is closed under taking finite extensions, and the fact that linear groups are residually finite, we obtain:

Corollary 7.7. *A GBS group $G = \pi_1(\mathcal{G})$ is linear if and only if it is a solvable Baumslag-Solitar group or if every cycle $t_1, \ldots, t_n$ in the graph Y satisfies*

$$\prod_{i=1}^{n} \frac{\beta_{t_i}}{\alpha_{t_i}} = \pm 1.$$

The proof of [Lev14, Cor. 7.7] works by embedding a non-residually finite Baumslag-Solitar group into a GBS group G if G is not solvable or virtually $F \times \mathbb{Z}$. If the requirements of Corollary 7.7 are met and G is not solvable, it is easy to construct a subgroup $F \times \mathbb{Z}$ of finite index. Indeed, let $M = \prod_{t \in E(Y)} \alpha_t \cdot \beta_t$ and let $a \in V(Y)$ be some vertex. Then the subgroup $\left\langle a^M \right\rangle \leq G$ is normal. Moreover, the subgroup of G generated by the edges of Y acts on $\left\langle a^M \right\rangle$ by inversion; hence, it has a subgroup F of index two stabilizing $\left\langle a^M \right\rangle$. It is straightforward to see that $F \times \left\langle a^M \right\rangle$ has finite index in G.

7.3. The Word Problem of GBS Groups

In [Rob93], Robinson showed that the word problem in non-cyclic free groups is NC^1-hard. Hence, for non-solvable GBS groups, we cannot expect the word or conjugacy problem to be in TC^0 since they contain a free group of rank two. Before considering the conjugacy problem of GBS groups, we have to deal with the word problem. For ordinary Baumslag-Solitar groups, the word problem has been shown to be in NC^2 [J. Kausch, private conversation]. We improve this result to $\mathsf{LOGDCFL}$ by giving a construction for a DLPAuxPDA which decides the word problem of a GBS group.

In the previous section, we have established how to distinguish between linear GBS groups and non-linear GBS groups. For linear groups, we can use the following result which was obtained by Lipton and Zalcstein [LZ77] for fields of characteristic 0 and by Simon [Sim79] for other fields.

Theorem 7.8 ([LZ77, Sim79])**.** *Linear groups have word problem in* $\mathsf{LOGSPACE}$.

Now, let us focus on arbitrary GBS groups. Let $G = \pi_1(\mathcal{G})$ be a fixed GBS group given by a graph Y and numbers $\alpha_t, \beta_t \in \mathbb{Z} \setminus \{0\}$ for $t \in E(Y)$. Recall that $\alpha_t = \beta_{\bar{t}}$ and we have

$$G \leq \left\langle V(Y), E(Y) \;\middle|\; tb^{\beta_t}\bar{t} = a^{\alpha_t}, \bar{t}t = 1 \text{ for } t \in E(Y), a = \iota(t), b = \tau(t) \right\rangle = F(\mathcal{G}).$$

As usual when considering conjugacy, it is more convenient to work outside of G in the group $F(\mathcal{G})$ in order to apply the Conjugacy Criterion. However, we need to consider $\mathcal{G}$-factorizations $w \in \Pi(\mathcal{G})$, only. Recall that a $\mathcal{G}$-factorization of some group element is a word $w = a_0^{k_0} t_1 a_1^{k_1} \cdots t_n a_n^{k_n}$ with $a_i = \iota(t_{i+1})$ for $0 \leq i < n$, $a_n = \tau(t_n) = a_0$, and $k_i \in \mathbb{Z}$. In the following, we always write a_i instead of $\iota(t_{i+1})$ in order to simplify notation.

We say that w is a *binary $\mathcal{G}$-factorization* or w is represented in *binary* if the numbers k_i are written in binary. Note that the binary representation for solvable Baumslag-Solitar groups can be viewed as a special case of this binary representation.

Lemma 7.9. *Let $w = a_0^{k_0} t_1 a_1^{k_1} \cdots t_n a_n^{k_n} \in \Pi(\mathcal{G})$. If $w \in \langle a_0 \rangle$ in $F(\mathcal{G})$, then we have $w =_{F(\mathcal{G})} a_0^k$ for*

$$k = \sum_{\nu=0}^{n} k_\nu \cdot \prod_{\mu=1}^{\nu} \frac{\alpha_\mu}{\beta_\mu}$$

where $\alpha_\mu = \alpha_{t_\mu}$ and $\beta_\mu = \beta_{t_\mu}$ for $1 \leq \mu \leq n$.

Proof. If $w = a_0^{k_0}$, then the formula is obviously correct. Hence, let $n > 0$. Then by Lemma 3.13, all the edges t_i can be cancelled by Britton reductions. In particular, t_1 cancels with some t_i, i. e., we can find some $1 < i \leq n$ such that $w = a_0^{k_0} t_1 w' t_i w''$ with $t_1 = \bar{t}_i$ and $w' = a_1^{k_1} t_2 \cdots a_{i-1}^{k_{i-1}} \in_{F(\mathcal{G})} \langle a_1 \rangle$ and $w' = a_i^{k_i} t_{i+1} \cdots a_n^{k_n} \in_{F(\mathcal{G})} \langle a_0 \rangle = \langle a_i \rangle$.

By induction, we have $w' =_{F(\mathcal{G})} a_1^{k'}$ and $w'' =_{F(\mathcal{G})} a_0^{k''}$ where

$$k' = \sum_{\nu=1}^{i-1} k_\nu \cdot \prod_{\mu=2}^{\nu} \frac{\alpha_\mu}{\beta_\mu}, \qquad k'' = \sum_{\nu=i}^{n} k_\nu \cdot \prod_{\mu=i+1}^{\nu} \frac{\alpha_\mu}{\beta_\mu}.$$

Since $t_1 w' t_i \in \langle a_0 \rangle$, we have $t_1 w' t_i =_{F(\mathcal{G})} a_0^{\frac{\alpha_1}{\beta_1} \cdot k'}$ and $\prod_{\mu=1}^{i} \frac{\alpha_\mu}{\beta_\mu} = 1$. Hence,

$$k = k_0 + \frac{\alpha_1}{\beta_1} k' + k'' = \sum_{\nu=0}^{n} k_\nu \cdot \prod_{\mu=1}^{\nu} \frac{\alpha_\mu}{\beta_\mu}$$

holds. □

For the rest of this section, we let $w = a_0^{k_0} t_1 a_1^{k_1} \cdots t_n a_n^{k_n}$ be a $\mathcal{G}$-factorization given in binary. For $0 \leq i \leq j \leq n$, we define

$$k_{i,j} = \sum_{\nu=i}^{j} k_\nu \cdot \prod_{\mu=i+1}^{\nu} \frac{\alpha_\mu}{\beta_\mu} \tag{7.3}$$

analogously to k in Lemma 7.9 (note that we do not assume that $a_i^{k_i} t_{i+1} a_{i+1}^{k_{i+1}} \cdots t_j a_j^{k_j}$ lies in $\langle a_i \rangle$ – yet these numbers will play an important part in what follows). In particular, with the notation of Lemma 7.9, we have $k = k_{0,n}$.

In the following, let $\mathcal{P}$ denote the set of all prime divisors of the α_t and β_t for $t \in E(Y)$.

Remark 7.10. If $k, \ell \in \mathbb{N}$ and ℓ only has prime divisors in $\mathcal{P}$, then $\gcd(k, \ell)$ can be computed in TC^0 by checking for all $p \in \mathcal{P}$ and all $e \in \{1, \ldots, \lfloor \log \ell \rfloor\}$ in parallel whether p^e divides both k and ℓ (using Integer Division, see Theorem 2.4).

This contrasts to the situation where ℓ might have arbitrary prime divisors; then it is even not known whether the gcd can be computed in NC.

Lemma 7.11. *The numbers $k_{i,j}$ (as reduced fractions of integers) can be computed by a uniform family of TC^0 circuits.*

Proof. Iterated Addition and Iterated Multiplication are in TC^0, see Theorem 2.4; hence, the rational numbers $k_{i,j}$ can be computed. Moreover, by Remark 7.10, reducing the fractions is possible in TC^0 since only powers of prime numbers in $\mathcal{P}$ might occur as prime factors of the denominator. □

Now we are ready to describe a DLPAuxPDA for the word problem in GBS groups. Recall that the input is some $\mathcal{G}$-factorization $w = a_0^{k_0} t_1 a_1^{k_1} \cdots t_n a_n^{k_n}$ represented in binary. If $w =_{F(\mathcal{G})} 1$, then w can be reduced to the empty word by only applying Britton reductions as in Section 3.2. We describe a deterministic logspace polynomial-time auxiliary pushdown automaton (DLPAuxPDA) which uses this fact in order to decide the word problem for $\mathcal{G}$-factorizations in $F(\mathcal{G})$. Sudborough [Sud77] showed that the languages recognized by DLPAuxPDAs are exactly $\mathsf{LOGDCFL}$, see Theorem 2.6.

The DLPAuxPDA reads the input from left to right. Letters $a_i^{k_i}$ are simply ignored, i. e., we assume that the read-head is always on some letter $t_j \in E(Y)$. When a letter $t_j \in E(Y)$ is read, either this letter is pushed onto the stack together with its index j, or the letter on the stack top is removed from the stack (together with its index). Which of these possibilities applies, depends on whether a Britton reduction canceling the two letters may occur. Then the read-head is moved to t_{j+1}. In particular, on the stack, there is always a sequence $(t_{i_1}, i_1), \ldots, (t_{i_m}, i_m)$ of edges t_{i_ℓ} together with their respective indices i_ℓ (for $1 \leq \ell \leq m$). The push operations can be performed by the DLPAuxPDA because the indices fit into the logarithmic working tape.

More precisely, the automaton works as follows: Let (t_i, i) be the stack top. When a letter t_j is read, (t_i, i) is popped from the stack top if and only if $t_i = \bar{t}_j$ and $k_{i,j-1} \in \beta_i \mathbb{Z}$ (recall the definition (7.3) of $k_{i,j-1}$). Otherwise, and in the case that the stack is empty, (t_j, j) is pushed onto the stack. By Lemma 7.11, the numbers $k_{i,j-1}$ can be computed without using the stack. Hence, the test whether $k_{i,j-1} \in \beta_i \mathbb{Z}$ can be done in logarithmic space by calculating modulo the constant number β_i.

If the stack is empty after reading the whole input, we have $w =_{F(\mathcal{G})} 1$ if and only if $k = k_{0,n} = 0$. If at the end the stack is not empty, the input w is not equal to 1 in $F(\mathcal{G})$. In Algorithm 7.1, we give the pseudocode of this algorithm.

Algorithm 7.1 *Word Problem of GBS groups*

function WordProblemGBS($w = a_0^{k_0} t_1 a_1^{k_1} \cdots t_n a_n^{k_n} \in \Pi(\mathcal{G})$)
begin
  for $j \leftarrow 1$ **to** n **do**
    if (stack is empty) **then**
      push((t_j, j))
    else
      $(t, i) \leftarrow$ stack-top
      if ($t = \bar{t}_j$ **and** $k_{i,j-1} \in \beta_i \mathbb{Z}$) **then** pop
      else push((t_j, j))
    endif
  endfor
  if (stack is empty **and** $k_{0,n} = 0$) **then return** "$= 1$"
  else return "$\neq 1$"
end

The stack is initially empty. With "pop" we mean that the stack top is removed, with "stack-top" the stack top is read, but not removed. The numbers $k_{i,j-1}$ can be computed "on the fly" or by a preliminary LOGSPACE reduction.

The correctness of this algorithm is straightforward by showing the following invariants which hold during the whole execution of the algorithm. Here, $(t_{i_1}, i_1), \ldots, (t_{i_m}, i_m)$ is the sequence of letters, together with their indices, on the stack and $t_j \in E(Y)$ with index j is the input symbol that is currently being read. We set $i_m = 0$ if the stack is empty, and $j = n + 1$ if the end of the input is reached:

$$a_0^{k_{0,i_1-1}} t_{i_1} a_{i_1}^{k_{i_1,i_2-1}} \cdots t_{i_m} a_{i_m}^{k_{i_m,j-1}} \quad \text{is Britton-reduced,} \tag{7.4}$$

$$a_0^{k_{0,i_1-1}} t_{i_1} a_{i_1}^{k_{i_1,i_2-1}} \cdots t_{i_m} a_{i_m}^{k_{i_m,j-1}} =_{F(\mathcal{G})} a_0^{k_0} t_1 a_1^{k_1} \cdots t_{j-1} a_{j-1}^{k_{j-1}}, \tag{7.5}$$

and

$$a_{i_m}^{k_{i_m,j-1}} =_{F(\mathcal{G})} a_{i_m}^{k_{i_m}} t_{i_m+1} a_{i_m+1}^{k_{i_m+1}} \cdots t_{j-1} a_{j-1}^{k_{j-1}}, \tag{7.6}$$

The invariants (7.4) and (7.5) together imply immediately the correctness of the algorithm.

In order to see these invariants, let $i = i_m$ be the index of the letter at the stack top and let the current letter be t_j at position j. If $j = 1$, then the stack is empty and all invariants hold trivially. Thus, let $j > 1$. By induction, we may assume that $a_0^{k_{0,i_1-1}} t_{i_1} a_{i_1}^{k_{i_1,i_2-1}} \cdots t_{i_m} a_{i_m}^{k_{i_m,j-1}}$ is Britton-reduced and $a_i^{k_i} t_{i+1} a_{i+1}^{k_{i+1}} \cdots t_{j-1} a_{j-1}^{k_{j-1}} =_{F(\mathcal{G})} a_i^{k_{i,j-1}}$. Let us distinguish the two cases that can occur:

- If t_i is removed from the stack, then we have $t_i = \bar{t}_j$ and $k_{i,j-1} \in \beta_i \mathbb{Z}$. The latter is, by induction hypothesis, equivalent to $a_i^{k_i} t_{i+1} a_{i+1}^{k_{i+1}} \cdots t_{j-1} a_{j-1}^{k_{j-1}} \in \left\langle a_i^{\beta_i} \right\rangle$.

Altogether, this means $a_{i-1}^{k_{i-1}} t_i a_i^{k_i} \cdots t_j a_j^{k_j} \in \langle a_{i-1} \rangle$ (i. e., t_i and t_j cancel in a Britton reduction). Hence, (7.6) holds by Lemma 7.9. As the stack has become shorter, obviously (7.4) holds, also. Now, (7.5) follows from (7.6) and the induction hypothesis.

- If t_j is pushed onto the stack, then we have $t_i \neq \bar{t}_j$ or $k_{i,j-1} \notin \beta_i \mathbb{Z}$. In either case, $t_i a_i^{k_{i,j-1}} t_j$ is Britton-reduced; hence, (7.4) holds by induction. Moreover, (7.6) holds because $a_j^{k_j} = a_j^{k_{j,j}}$. Again, (7.5) follows from (7.6) and the induction hypothesis.

Hence, we have shown that the word problem for $\mathcal{G}$-factorizations in $F(\mathcal{G})$ can be solved in LOGDCFL. In particular, we have:

Theorem 7.12. *All generalized Baumslag-Solitar groups have word problem in* LOGDCFL *– even if the input is given as binary $\mathcal{G}$-factorization.*

Lemma 7.13. *Assume that the word problem of a GBS group $\pi_1(\mathcal{G})$ can be solved in some complexity class $\mathcal{C}$ when the input is a $\mathcal{G}$-factorizations given in binary. Then, the following can be done in* $\mathsf{LOGSPACE}^{\mathcal{C}}$*:*

(i) *Given a $\mathcal{G}$-factorization w in binary, calculate a Britton-reduced $\mathcal{G}$-factorization $\hat{w}$ such that $w =_{F(\mathcal{G})} \hat{w}$.*

(ii) *Given a $\mathcal{G}$-factorization w in binary, calculate a cyclically reduced $\mathcal{G}$-factorization $\hat{w}$ such that $w \sim_{F(\mathcal{G})} \hat{w}$.*

Proof. Again, let $w = a_0^{k_0} t_1 a_1^{k_1} \cdots t_n a_n^{k_n} \in \Pi(\mathcal{G})$. We describe a LOGSPACE algorithm with oracle calls to the word problems of $\pi_1(\mathcal{G}, a)$ for $a \in V(Y)$. The algorithm follows the idea in [DKL12, DK14] for reductions in right-angled Coxeter groups and graph groups. By Lemma 7.11, the numbers $k_{i,j} = \sum_{\nu=i}^{j} k_\nu \cdot \prod_{\mu=i+1}^{\nu} \frac{\alpha_\mu}{\beta_\mu}$ can be computed in logarithmic space.

At the beginning, a variable ℓ storing an index is initialized with 0. The algorithm scans the input from left to right and increases an index i starting with $i = 0$. In every step, the maximal $j > i$ with $a_i^{k_i} t_{i+1} a_{i+1}^{k_{i+1}} \cdots t_j a_j^{k_j} =_{F(\mathcal{G})} a_i^{k_{i,j}}$ is determined by at most n calls to the word problem. If there is such j, then i is set to j. If no such j exists at all, then $a_i^{k_{\ell,i}} t_{i+1}$ is written on the output tape, ℓ is set to $i+1$, and i is increased by one. Obviously the output $\hat{w}$ is Britton-reduced and $w =_{F(\mathcal{G})} \hat{w}$.

Now, let $w = a_0^{k_0} t_1 a_1^{k_1} \cdots t_n a_n^{k_n}$ be Britton-reduced. We want to reduce w cyclically. If $n = 0$, then w is already cyclically reduced. Otherwise, we may assume $k_0 = 0$. We check the two cases of Lemma 3.18: By applying Britton reductions to ww, we obtain a Britton-reduced word $t_1 a_1^{k_1} \cdots t_{n-m} a_{n-m}^{\tilde{k}} t_{m+1} a_{m+1}^{k_{m+1}} \cdots t_n a_n^{k_n}$ for some $\tilde{k} \in \mathbb{Z}$, $m \in \mathbb{N}$. Depending on m, either $t_{m+1} a_{m+1}^{k_{m+1}} \cdots t_{n-m} a_{n-m}^{\tilde{k}}$ or a Britton reduced word for $t_{n/2+1} a_{n/2+1}^{k_{n/2+1}} \cdots t_n a_n^{k_n}\, t_1 a_1^{k_1} \cdots t_{n/2} a_{n/2}^{k_{n/2}}$ is the $\hat{w}$ we are looking for. □

By Theorem 2.5, we have $\mathsf{LOGSPACE}^{\mathsf{LOGDCFL}} = \mathsf{LOGDCFL}$. Hence, from Theorem 7.12 and Lemma 7.13, we obtain:

Corollary 7.14. *Let $G = \pi_1(\mathcal{G})$ be a GBS group. Then, the following can be done in* LOGDCFL. *Moreover, if G is linear, it can be done in* LOGSPACE.

- *Given a $\mathcal{G}$-factorization w in binary, calculate a Britton-reduced $\mathcal{G}$-factorization $\hat{w}$ such that $w =_{F(\mathcal{G})} \hat{w}$.*
- *Given a $\mathcal{G}$-factorization w in binary, calculate a cyclically reduced $\mathcal{G}$-factorization $\hat{w}$ such that $w \sim_{F(\mathcal{G})} \hat{w}$.*

7.4. The Conjugacy Problem of GBS Groups

According to Corollary 5.12, the conjugacy problem in GBS groups is decidable (note, however, that the proof of Corollary 5.12 relies on the current section). First, this result was shown by Anshel [Ans76a] in the special case where the graph Y consists of only one vertex. Later in [Hor84], Horadam showed that the conjugacy problem is decidable in GBS groups if there is some constant $c \in \mathbb{Z}$ with $\alpha_t = c$ for all $t \in E(Y)$. In [HF94], this was further generalized to some other class of GBS groups which contains the linear GBS groups (with the generalization that they also considered infinite graphs); in [Loc92], Lockhart gave a solution for all GBS groups. Finally, in [Bee11b], Beeker independently gave a solution of the conjugacy problem in all GBS groups.

However, in all these papers no bound for the complexity is given. Here, we take a closer look at all steps of the decision algorithm and show that, for cyclically reduced words, it is in LOGSPACE.

Lemma 7.15. *Let $\mathcal{P}$ be some fixed finite set of prime numbers. The following problem is solvable in TC^0: Given $c_i, d_i \in \mathbb{Z}$ (in binary) for $i = 0, \dots, n$ such that d_i has only prime factors in $\mathcal{P}$. Decide whether the system of congruences*

$$x \equiv c_i \mod d_i \qquad \text{for } i = 0, \dots, n$$

has a solution.

Proof. Since $\mathcal{P}$ is finite, the following can be done for all $p \in \mathcal{P}$ in parallel. Considering only powers of p, the system of congruences transforms into

$$x \equiv c_i \mod p^{e_i} \qquad \text{for } i = 0, \dots, n \tag{7.7}$$

where e_i is maximal such that p^{e_i} divides d_i. Such e_i can be determined in TC^0 by checking whether p^e divides d_i for all $0 \leq e \leq \log |d_i|$ in parallel using Integer Division, Theorem 2.4. If there is some $i \neq j$ with $e_i \leq e_j$ and $c_i \not\equiv c_j \mod p^{e_i}$, then (7.7) obviously does not have a solution. Again this can be checked in parallel for all pairs i, j. If there is no such pair $i \neq j$, (7.7) is equivalent to a single congruence $x \equiv c \mod p^e$ where $e = \max_{i \in \{0,\dots,n\}} e_i$ and $c = c_i$ for the respective i.

If (7.7) has a solution for all $p \in \mathcal{P}$, then there is a solution for the original congruence by the Chinese Remainder Theorem. □

Proposition 7.16. *The following problem is in* TC^0*: Given two cyclically reduced* $\mathcal{G}$*-factorizations* $v, w \in \Pi(\mathcal{G})$ *in binary representation such that* v *and* w *cannot be conjugated into any* $\langle a \rangle$ *for* $a \in V(Y)$*, decide whether* $v \sim_{F(\mathcal{G})} w$*.*

The proof of Proposition 7.16 follows the same idea as the proofs of Theorem 5.10 and Proposition 5.13; however, by a more careful inspection we can derive a good complexity bound in the special case of generalized Baumslag-Solitar groups.

Proof. By assumption, we are in case (iii) of the Conjugacy Criterion, Theorem 3.19. Let

$$v = t_1 a_1^{k_1} \cdots t_n a_n^{k_n}$$

be a $\mathcal{G}$-factorization. By Theorem 3.19 (iii) we know that if v and w are conjugate, then the underlying path $t_1 \cdots t_n$ of v is a cyclic permutation of the underlying path of w. Since in TC^0 all these cyclic permutation can be checked in parallel, we may assume that w is of the form

$$w = t_1 a_1^{\ell_1} \cdots t_n a_n^{\ell_n}.$$

and (also by Theorem 3.19 (iii))

$$v \sim_{F(\mathcal{G})} w \iff \exists\, x \in \mathbb{Z} \text{ such that } a^x v a^{-x} =_{F(\mathcal{G})} w.$$

By Lemma 3.13 (iv), we have $a^x v a^{-x} =_{F(\mathcal{G})} w$ if and only if there are $y_1, \ldots, y_n \in \mathbb{Z}$ such that

$$\begin{aligned} x - \alpha_1 y_1 &= 0, \\ \beta_i y_i + k_i - \alpha_{i+1} y_{i+1} &= \ell_i && \text{for } i = 1, \ldots, n-1, \\ \beta_n y_n + k_n - x &= \ell_n. \end{aligned}$$

As in [Hor84], these equations imply that it can be decided whether v and w are conjugate. As we aim for a good complexity bound, we have to take a closer look. By solving these equations for y_{i+1}, we obtain

$$\begin{aligned} y_1 &= \frac{x}{\alpha_1}, \\ y_{i+1} &= \frac{k_i - \ell_i + \beta_i y_i}{\alpha_{i+1}} && \text{for } i = 1, \ldots, n-1, \\ x &= k_n - \ell_n + \beta_n y_n. \end{aligned}$$

By induction follows

$$y_i = \frac{1}{\alpha_i} \left(x \cdot \prod_{\mu=1}^{i-1} \frac{\beta_\mu}{\alpha_\mu} + \sum_{\nu=1}^{i-1} (k_\nu - \ell_\nu) \cdot \prod_{\mu=\nu+1}^{i-1} \frac{\beta_\mu}{\alpha_\mu} \right) \qquad \text{for } i = 1, \ldots, n, \tag{7.8}$$

and the last equation becomes

$$x = k_n - \ell_n + x \cdot \prod_{\mu=1}^{n} \frac{\beta_\mu}{\alpha_\mu} + \sum_{\nu=1}^{n-1} (k_\nu - \ell_\nu) \cdot \prod_{\mu=\nu+1}^{n} \frac{\beta_\mu}{\alpha_\mu}. \tag{7.9}$$

We distinguish two cases:

First, assume that (7.9) has a unique solution. Then, the values y_i are also determined uniquely and we have $v \sim_{F(\mathcal{G})} w$ if and only if x and the y_i are all integers. In this case, we have $\prod_{\mu=1}^{n} \frac{\beta_\mu}{\alpha_\mu} \neq 1$ and

$$x = \frac{\sum_{\nu=1}^{n} (k_\nu - \ell_\nu) \cdot \prod_{\mu=\nu+1}^{n} \frac{\beta_\mu}{\alpha_\mu}}{1 - \prod_{\mu=1}^{n} \frac{\beta_\mu}{\alpha_\mu}}.$$

All occurring numbers are rationals; hence, they can be represented as fractions of binary integers. Since Iterated Multiplication is in TC^0 (Theorem 2.4), the products can be computed. A common denominator for the sums can be computed by Iterated Multiplication, again. Thus, calculating the sum is just Iterated Addition (Theorem 2.4). Let $c, d, e, f \in \mathbb{Z}$ be such that $\frac{c}{d} = \sum_{\nu=1}^{n} (k_\nu - \ell_\nu) \cdot \prod_{\mu=\nu+1}^{n} \frac{\beta_\mu}{\alpha_\mu}$ and $\frac{e}{f} = 1 - \prod_{\mu=1}^{n} \frac{\beta_\mu}{\alpha_\mu}$. In case $x = \frac{cf}{de}$ is an integer, we can determine this by applying Hesse's circuit for Integer Division (Theorem 2.4) to cf and de. If x is not an integer, we can notice that by multiplying the result of the division with de; if the result is not cf, there is no x with $a^x v a^{-x} =_{F(\mathcal{G})} w$.

If x is an integer, the numbers y_i can be computed in TC^0 with the same technique, and it can be checked whether $y_i \in \mathbb{Z}$ for all i. Thus, we are done with the case that (7.9) has a unique solution.

In the second case, we have $\prod_{\mu=1}^{n} \frac{\beta_\mu}{\alpha_\mu} = 1$. Then (7.9) is equivalent to

$$k_n - \ell_n + \sum_{\nu=1}^{n-1} (k_\nu - \ell_\nu) \cdot \prod_{\mu=\nu+1}^{n} \frac{\beta_\mu}{\alpha_\mu} = 0.$$

Again, this equality can be checked in TC^0 as before. If the equality does not hold, then there is no x with $a^x v a^{-x} =_{F(\mathcal{G})} w$. Otherwise, by (7.8), we have $a^x v a^{-x} =_{F(\mathcal{G})} w$ for $x \in \mathbb{Z}$ if and only if

$$\frac{1}{\alpha_i} \left(x \cdot \prod_{\mu=1}^{i-1} \frac{\beta_\mu}{\alpha_\mu} + \sum_{\nu=1}^{i-1} (k_\nu - \ell_\nu) \cdot \prod_{\mu=\nu+1}^{i-1} \frac{\beta_\mu}{\alpha_\mu} \right) \in \mathbb{Z} \qquad \text{for all } i \in \{1, \ldots, n\}.$$

By solving for x, we obtain

$$x \in \mathbb{Z} \cap \bigcap_{i=1}^{n} \left(\alpha_i \cdot \prod_{\mu=1}^{i-1} \frac{\alpha_\mu}{\beta_\mu} \right) \cdot \left(-\frac{1}{\alpha_i} \sum_{\nu=1}^{i-1} (k_\nu - \ell_\nu) \cdot \prod_{\mu=\nu+1}^{i-1} \frac{\beta_\mu}{\alpha_\mu} + \mathbb{Z} \right). \tag{7.10}$$

Let M be a common denominator of all terms in this sum and $c_i, d_i \in \mathbb{Z}$ such that

$$\frac{c_i}{M} = -\left(\prod_{\mu=1}^{i-1} \frac{\alpha_\mu}{\beta_\mu}\right) \cdot \sum_{\nu=1}^{i-1} (k_\nu - \ell_\nu) \cdot \prod_{\mu=\nu+1}^{i-1} \frac{\beta_\mu}{\alpha_\mu} \qquad \text{and}$$

$$\frac{d_i}{M} = \alpha_i \cdot \prod_{\mu=1}^{i-1} \frac{\alpha_\mu}{\beta_\mu} \qquad \text{for } i = 1, \ldots, n.$$

In addition, we set $c_0 = 0$ and $d_0 = M$. Now, (7.10) is equivalent to

$$x \in \frac{1}{M}\left(c_i + d_i\mathbb{Z}\right) \qquad \text{for all } i \in \{0, \ldots, n\}.$$

We set $y = Mx$. Because of the choice of c_0 and d_0, the existence of an integer solution x is equivalent to the system of congruences

$$y \equiv c_i \mod d_i \qquad \text{for } i = 0, \ldots, n \tag{7.11}$$

having a solution. Again, let $\mathcal{P}$ be the finite set of prime divisors of the α_t and β_t for $t \in E(Y)$. As M as well as the d_is are products of the α_t and β_t, they have only prime factors in $\mathcal{P}$. Furthermore, as before, the numbers c_i, d_i and M can be computed in TC^0. Now, by Lemma 7.15, it can be checked in TC^0 whether (7.11) has a solution. □

Proposition 7.17. *The following problem is in* TC^0*: Given* $v = a^k$ *and* $w = a^\ell$ *with* $k, \ell \in \mathbb{Z}$*, decide whether* $v \sim_{\mathbf{BS}_{p,q}} w$*.*

Proof. We are in case (i) and (ii) of the Conjugacy Criterion, Theorem 3.19. Since a conjugation with a has no effect and a conjugation with $t^{\pm 1}$ multiplies the exponent by $\frac{p}{q}$ resp. $\frac{q}{p}$, we have

$$a^k \sim_{\mathbf{BS}_{p,q}} a^\ell \iff \exists\, j \in \mathbb{Z} \text{ such that } k \cdot \left(\frac{p}{q}\right)^j = \ell.$$

Since we have $|j| \leq \log_{|q/p|} \max\{|k|, |\ell|\}$ if such j exists, only polynomially many (in the input size) values for j need to be tested, what can be done in parallel. As Iterated Multiplication and Integer Division are in TC^0 ([Hes01, HAB02], see Theorem 2.4), we have concluded the proof of Proposition 7.17. □

Note that Proposition 7.3 (conjugacy for solvable Baumslag-Solitar groups) can be derived as a special case of Proposition 7.16 and Proposition 7.17 (however, the proof of Proposition 7.3 is much easier). Also, for non-solvable Baumslag-Solitar groups, we have solved the conjugacy problem completely by combining Corollary 7.14 with Proposition 7.16 and Proposition 7.17.

For arbitrary generalized Baumslag-Solitar groups, it remains to examine cases (i) and (ii) of Theorem 3.19. These cases are also needed to conclude the proof of Theorem 5.10. Hence, let again $G = \pi_1(\mathcal{G})$ be some GBS group. We follow the ideas of Anshel [Ans76a, Ans76b, Ans76c] in order to describe a $\mathsf{LOGSPACE}$ algorithm for the conjugacy

problem in this case. Let $v = a^k$, $w = b^\ell$ for some $a, b \in V(Y)$. By Theorem 3.19, we know that $v \sim_{F(\mathcal{G})} w$ if and only if there is some $x = a_0^{k_0} t_1 a_1^{k_1} \cdots t_n a_n^{k_n} \in \Pi(\mathcal{G}, b, a)$ (as before, $a_i = \iota(t_{i+1})$ for $i = 0, \ldots, n-1$ and $a_n = \tau(t_n)$) such that $xa^k x^{-1} =_{F(\mathcal{G})} w$ and

$$t_i a_i^{k_i} \cdots t_n\, a_n^{k_n} \cdot a^k \cdot a_n^{-k_n} \bar{t}_n \cdots a_i^{-k_i} \bar{t}_i \in G_{t_i}^{t_i} \qquad \text{for all } i.$$

Since a conjugation with a_i has no effect on elements of $G_{a_i} = \langle a_i \rangle$, we may assume that $x = t_1 \cdots t_n$ if v and w are conjugate.

Let $\mathcal{P} = \{p_1, \ldots, p_m\}$ as before be the set of prime divisors occurring in the α_t for $t \in E(Y)$. Here and in what follows, we treat -1 as a prime number. Let

$$k = r_k \cdot \prod_{i=1}^{m} p_i^{e_i(k)}, \qquad \ell = r_\ell \cdot \prod_{i=1}^{m} p_i^{e_i(\ell)}, \tag{7.12}$$

such that $r_k, r_\ell > 0$ are not divisible by any $p \in \mathcal{P} \setminus \{-1\}$. The numbers r_k, r_ℓ and the exponents $e_i(k), e_i(\ell)$ can be computed in TC^0 (in particular, in $\mathsf{LOGSPACE}$) as before by checking for all $p \in \mathcal{P}$ and $e \leq \log |k|$ whether p^e divides k using Hesse's TC^0 circuit for Integer Division, Theorem 2.4 (for $p = -1$, it has to be checked whether $k > 0$) – and likewise for ℓ. If $v \sim_{F(\mathcal{G})} w$, then $r_k = r_\ell$. Hence, it remains to consider the vectors $(e_1(k), \ldots, e_m(k)), (e_1(\ell), \ldots, e_m(\ell)) \in \mathbb{N}^m$.

Let $e = (e_1, \ldots, e_m), f = (f_1, \ldots, f_m) \in \mathbb{N}^m$ be some arbitrary vectors and let $\tilde{a} \in V(\Gamma)$. We define $e \sim_{\tilde{a}} f$ if

$$\tilde{a}^{\prod_{i=1}^m p_i^{e_i}} \sim_{F(\mathcal{G})} \tilde{a}^{\prod_{i=1}^m p_i^{f_i}}.$$

The numbers $e_i(k), e_i(\ell)$ of (7.12) are bounded by a linear function in the input size. In particular, if $a = b$, then we have a $\mathsf{LOGSPACE}$ reduction from the question whether $a^k \sim_{F(\mathcal{G})} a^\ell$ to the question whether $(e_1(k), \ldots, e_m(k)) \sim_a (e_1(\ell), \ldots, e_m(\ell))$ where the numbers $e_i(k), e_i(\ell)$ are represented in unary. Thus, we aim for a $\mathsf{LOGSPACE}$ algorithm to decide whether $e \sim_a f$ for vectors $e, f \in \mathbb{N}^m$ given in unary.

The following crucial fact is an immediate consequence of the definition of $\sim_{\tilde{a}}$.

Lemma 7.18. *If $e \sim_{\tilde{a}} f$, then also $e + g \sim_{\tilde{a}} f + g$ for all $g \in \mathbb{N}^m$.*

Hence, $\sim_{\tilde{a}}$ defines a congruence on $\mathbb{N}^m$. Thus, $\mathbb{N}^m / \sim_{\tilde{a}}$ is a monoid and our task is to solve the word problem of this monoid (the word problem of a monoid is the question whether two given elements are equal in the monoid). The result that the word problem for finitely generated commutative monoids is decidable, is attributed to Malcev [Mal58] and Emelichev [Eme58]. According to [Car75, MM82], there are several results implying that the word problem is solvable with a real-time Turing machine [Bir67, FMR68, GS66, Lai67, Taĭ68]. In [Pop02], an explicit proof can be found.

In [BL81], Ballantyne and Lankford showed how to construct a Church-Rosser rewriting system in order to solve the uniform word problem of f. g. commutative monoids (i. e., where the congruence is part of the input). Here, we follow this approach in order to show that the word problem of a fixed f. g. commutative monoid can be solved in $\mathsf{LOGSPACE}$. We present a construction for a weight-reducing confluent rewriting system for a finitely generated commutative monoid.

Proposition 7.19. *Let $\sim$ be some arbitrary congruence on $\mathbb{N}^m$. On input of two vectors $e, f \in \mathbb{N}^m$ in unary, it can be decided in* LOGSPACE *whether $e \sim f$.*

Before we prove Proposition 7.19, we need some definitions. We define two different (partial) orders on $\mathbb{N}^m$. First, let

$$(e_1, \ldots, e_m) \leq (f_1, \ldots, f_m) \text{ if and only if } e_i \leq f_i \text{ for all } i \in \{1, \ldots, m\}.$$

For $S \subseteq \mathbb{N}^m$, we define $\min_{\leq}(S)$ as the set of minimal elements of S, i. e.,

$$\min_{\leq}(S) = \{e \in S \mid \text{there is no } f \in S \text{ with } f \leq e \text{ and } f \neq e\}.$$

The second order is the lexicographic order on $\mathbb{N}^m$, which we denote by $\prec$, i. e.,

$$\begin{aligned}(e_1, \ldots, e_m) \prec (f_1, \ldots, f_m) \text{ if and only if} \\ \text{there is some } i \text{ such that } e_i < f_i \text{ and } e_j = f_j \text{ for } j < i.\end{aligned}$$

Note that both orders are compatible with the addition in $\mathbb{N}^m$, i. e., $e \leq f$ implies $e + g \leq f + g$ for all $g \in \mathbb{N}^m$, and $e \prec f$ implies $e + g \prec f + g$ for all $g \in \mathbb{N}^m$. The following lemmata summarize some important standard facts about these orders.

Lemma 7.20. *The order $\prec$ is a well-order, i. e., it is total and there are no infinite strictly descending chains.*

Lemma 7.21 (Dickson's Lemma, [Dic13])**.** *Let $S \in \mathbb{N}^m$. Then $\min_{\leq}(S)$ is finite, and for every $e \in S$, there is some $f \in \min_{\leq}(S)$ with $f \leq e$.*

Proof of Proposition 7.19. We describe a finite convergent rewriting system $\mathcal{R} \subseteq \mathbb{N}^m \times \mathbb{N}^m$ on the monoid $(\mathbb{N}^m, +)$ such that $\overset{*}{\underset{\mathcal{R}}{\Longleftrightarrow}} = \sim$. First, we set

$$\begin{aligned}S &= \{e \in \mathbb{N}^m \mid \exists\, f \in \mathbb{N}^m : f \prec e \text{ and } f \sim e\}, \\ L &= \min_{\leq}(S).\end{aligned}$$

By Lemma 7.21, L is finite. For every $\ell \in L$, we fix some r_ℓ with $r_\ell \prec \ell$ and $r_\ell \sim \ell$. By definition of S, such r_ℓ always exists. Now,

$$\mathcal{R} = \{\ell \to r_\ell \mid \ell \in L\}$$

is a finite rewriting system. Recall the definition for rewriting in monoids as presented in Chapter 2: if $(\ell, r_\ell) \in \mathcal{R}$ is some rewriting rule, then it can be applied to a vector e if and only if $\ell \leq e$; the result of the rewriting step is $e - \ell + r_\ell$. By the definition of $\mathcal{R}$, it is clear that $e \overset{*}{\underset{\mathcal{R}}{\Longleftrightarrow}} f$ implies $e \sim f$. Let $e \in \mathbb{N}^m$ and let f be the $\prec$-smallest element in $\{g \in \mathbb{N}^m \mid g \sim e\}$ (such a smallest element exists and it is unique by Lemma 7.20). We show that $e \overset{*}{\underset{\mathcal{R}}{\Longrightarrow}} f$.

If $e \neq f$, then we have $e \in S$. Thus, by Lemma 7.21, there is some $\ell \in L$ with $\ell \leq e$ and it follows that $e \underset{\mathcal{R}}{\Longrightarrow} e - \ell + r_\ell \sim e$. Since $\prec$ is well-founded and $e - \ell + r_\ell \prec e$, we can apply induction and it follows that $e - \ell + r_\ell \overset{*}{\underset{\mathcal{R}}{\Longrightarrow}} f$. Thus, $e \overset{*}{\underset{\mathcal{R}}{\Longrightarrow}} f$.

Hence, the rewriting system $\mathcal{R}$ is convergent and $e \overset{*}{\underset{\mathcal{R}}{\iff}} f$ if and only if $e \sim f$ for $e, f \in \mathbb{N}^m$. It remains to show that for every $e = (e_1, \ldots, e_m) \in \mathbb{N}^m$ given in unary the smallest element $f \in \{g \in \mathbb{N}^m \mid g \sim e\}$ can be computed in LOGSPACE. Since the components e_i are bounded by the input length and m is a constant, the vector e can be stored in binary on the work tape. For $e \in \mathbb{N}^m$ we assign a weight $\gamma(e) = \sum_{i=1}^m e_i\gamma_i$ such that $\gamma_i > \max\left\{\sum_{j=i+1}^m \gamma_j (r_\ell)_j \;\middle|\; \ell \in L\right\}$ for all $1 \leq i \leq m$ (where $(r_\ell)_j$ denotes the j-th component of the vector r_ℓ). It follows that $\mathcal{R}$ is weight-reducing. Hence, when applying arbitrarily many rules of $\mathcal{R}$ to e, the resulting numbers are still bounded by some linear function in $\sum_{i=1}^m e_i$. Thus, the logarithmic space is enough to find the smallest element $f \in \{g \in \mathbb{N}^m \mid g \sim e\}$ by successively applying arbitrary rules of $\mathcal{R}$ to e. Hence, we have concluded the proof of Proposition 7.19. □

As $V(Y)$ is finite, there are only a fixed number of different congruences $\sim_{\tilde{a}}$ for $\tilde{a} \in V(Y)$. Therefore, by Proposition 7.19 and the above considerations, we have obtained a solution to the conjugacy problem for $v = a^k$, $w = b^\ell$ in the case that $a = b$. For the general case, we can use the same technique as before. In order to do so, we define for $a, b \in V(Y)$:

$$T_{a,b} = \min_{\leq}\left\{e \in \mathbb{N}^m \;\middle|\; a^{\prod_{i=1}^m p_i^{e_i}} \sim_{F(\mathcal{G})} b^\ell \text{ for some } \ell \in \mathbb{Z}\right\}.$$

Again, by Lemma 7.21, $T_{a,b}$ is finite. For every $e \in T_{a,b}$, we choose some $f = f_e \in \mathbb{N}^m$ such that $a^{\prod_{i=1}^m p_i^{e_i}} \sim_{F(\mathcal{G})} b^{\prod_{i=1}^m p_i^{f_i}}$ and set $\mathcal{R}_{a,b} = \{(e, f_e) \mid e \in T_{a,b}\}$. With this definition the next lemma is straightforward to see:

Lemma 7.22. *Let*

$$v = a^{r_k \cdot \prod_{i=1}^m p_i^{e_i(k)}}, \qquad w = b^{r_\ell \cdot \prod_{i=1}^m p_i^{e_i(\ell)}}.$$

Then $v \sim_{F(\mathcal{G})} w$ if and only if $r_k = r_\ell$ and there is some $(e, f_e) \in \mathcal{R}_{a,b}$ with $e \leq (e_1(k), \ldots, e_m(k))$ and $(e_1(k), \ldots, e_m(k)) - e + f_e \sim_b (e_1(\ell), \ldots, e_m(\ell))$.

The condition of Lemma 7.22 can be checked in LOGSPACE. Hence, we have finished the description of a LOGSPACE algorithm for deciding whether $a^k \sim_{F(\mathcal{G})} b^\ell$ for $a, b \in V(Y)$, $k, \ell \in \mathbb{Z}$. Now, we can combine this result with Corollary 7.14 and Proposition 7.16. Because, by Theorem 2.5, LOGDCFL is closed under $\mathsf{LOGSPACE}^{\mathsf{LOGDCFL}}$ reductions, we obtain a proof of the main result of this chapter (the solvable case is just a repetition of Theorem 7.4).

Theorem 7.23. *Let $G = \pi_1(\mathcal{G})$ be a generalized Baumslag-Solitar group. Then the conjugacy problem of G is in* LOGDCFL*. Moreover, if G is linear, it is in* LOGSPACE*, and if G is solvable, it is in* TC^0*.*

For a classification into these three classes of groups see Corollary 7.7 and Lemma 7.5.

7.5. The Uniform Conjugacy Problem

When the group G is not fixed anymore, the complexity of the conjugacy problem changes dramatically. The *uniform conjugacy problem* for GBS groups receives as input a graph of groups $\mathcal{G}$ consisting of a finite graph Y and numbers $\alpha_t, \beta_t \in \mathbb{Z} \setminus \{0\}$ for $t \in E(Y)$ and two $\mathcal{G}$-factorizations v, w. The question is whether $v \sim w$ in $\pi_1(\mathcal{G})$. In [AM83], Anshel and McAloon considered a special (more difficult) version of the uniform conjugacy problem; they showed that the so-called finite special equality problem for some GBS groups is decidable but not primitive recursive. However, they did not consider the general uniform conjugacy problem. Following the ideas we developed for the non-uniform case, we obtain a hardness result for the uniform conjugacy problem.

Theorem 7.24. *The uniform conjugacy problem for GBS groups is* EXPSPACE*-hard.*

The proof of Theorem 7.24 is an application of the next theorem by Cardoza, Lipton and Meyer [CLM76] (preliminary version), which was finally presented by Mayr and Meyer in [MM82].

Theorem 7.25 ([CLM76, MM82])**.** *The uniform word problem for finitely presented commutative semigroups is* EXPSPACE*-complete.*

We conjecture that also the uniform conjugacy problem for GBS groups is EXPSPACE-complete. However, given a graph of groups for a GBS group, it is not obvious how to find a finite set of relators which defines an instance for the word problem of finitely presented commutative semigroups. Another point is that it is not clear how to find the sets $T_{a,b}$ in the case that the group elements v, w are from different vertex groups.

Proof of Theorem 7.24. We give a LOGSPACE reduction from the uniform word problem of f. g. commutative semigroups to the uniform conjugacy problem for GBS groups. W. l. o. g. we only consider commutative monoids. Let $e, f \in \mathbb{N}^m, (r_i, s_i)_{i \in \{1 \dots k\}}$ with $r_i, s_i \in \mathbb{N}^m$ be some instance for the uniform word problem of commutative monoids (i. e., the question is whether $e \sim f$ where $\sim$ is the smallest congruence satisfying $r_i \sim s_i$ for all i). We construct an instance for the uniform conjugacy problem as follows: The graph Y consists of a single vertex; for all $i \in \{1, \dots, k\}$ there is an undirected edge $\{t_i, \bar{t}_i\} \subseteq E(Y)$. Let $\mathcal{P} = \{p_1, \dots, p_m\}$ be the set of the first m prime numbers. The numbers p_j can be computed in LOGSPACE since they require a logarithmic (in m) number of bits, only (by the prime number theorem there are enough primes, see e. g. [DKR13, (2.9)]). Now, for every relator (r_i, s_i), we define $\alpha_{t_i} = \prod_{j=0}^{m} p_j^{(r_i)_j}$ and $\beta_{t_i} = \prod_{j=0}^{m} p_j^{(s_i)_j}$ where $(r_i)_j$ denotes the jth component of the vector r_i. Likewise, we define $v = \prod_{j=0}^{m} p_j^{e_j}$ and $w = \prod_{j=0}^{m} p_j^{f_j}$. It is straightforward to see that $v \sim_{\pi_1(\mathcal{G})} w$ if and only if $e \sim f$. □

If we restrict ourselves to the class of GBS groups where the underlying graph has only one vertex, we can also show that the uniform conjugacy problem is in EXPSPACE.

Corollary 7.26. *The uniform conjugacy problem for GBS groups where the graph has only one vertex is* EXPSPACE*-complete.*

Proof. The hardness result follows from the proof of Theorem 7.24. On the other hand, an instance of the uniform conjugacy problem for GBS groups with one vertex in the underlying graph immediately gives rise to an instance of the uniform word problem of commutative semigroups, see Lemma 7.18. □

Chapter 8.

Conjugacy in the Baumslag Group

In this chapter, we want to deal with the conjugacy problem of the Baumslag group $\mathbf{G}_{1,2}$ (sometimes also called Baumslag-Gersten group, see [Pla04]). In 1969, Gilbert Baumslag introduced the group $\mathbf{G}_{1,2}$ as an example of an infinite non-cyclic group all of whose finite quotients are cyclic [Bau69]. In particular, it is not residually finite; but – being a one-relator group – it has a decidable word problem [Mag32].

The Baumslag group is an HNN extension of the Baumslag-Solitar group $\mathbf{BS}_{1,2}$. Hence, one can use Britton reductions (as in Section 3.2) to solve the word problem. However, by doing so, the lengths of the resulting words grow non-elementarily. In addition, Gersten showed that the Dehn function of $\mathbf{G}_{1,2}$ is non-elementary [Ger92], see also [Ger93]. Therefore, the group $\mathbf{G}_{1,2}$ was believed to have extremely difficult word problem until Myasnikov, Ushakov and Won showed in [MUW12] that it is actually in P. Their idea was to introduce a new data structure, so-called power circuits, in order to deal with the extremely large numbers which appear when applying Britton reductions. Power circuits are a way to compress such numbers efficiently. Moreover, they allow that all necessary arithmetic operations for solving the word problem can be performed in the compressed form.

Nevertheless, for the conjugacy problem in $\mathbf{G}_{1,2}$, the situation is different. In order to deal with conjugacy, one needs to decide the divisibility problem in power circuits – there is no method known to do this in elementary time.

The first part of this chapter introduces power circuits and examines the divisibility problem more carefully. Then, in Section 8.2, we show that the conjugacy problem of $\mathbf{G}_{1,2}$ is doubly strongly generically in P, although we do not know an elementary time algorithm for the worst case. The material presented in Section 8.2 has been published in [DMW14] before.

8.1. Power Circuits

In binary, a number is represented as a sum $m = \sum_{i=0}^{k} b_i 2^i$ with $b_i \in \{0, 1\}$. Allowing $b_i \in \{-1, 0, 1\}$, we obtain a "compact representation" of integers, which may require less non-zero b_is than the normal representation. Now, if m is a huge number and only a very small fraction of the b_is are non-zero, it still requires a lot of memory to store m, although there is very few information contained in it. This is where the idea of power circuits starts: instead of storing the whole binary number, only the positions i with $b_i \neq 0$ are stored – recursively using power circuits. In this way, huge numbers

like tower functions can be represented using only little memory. The notion of power circuit is due to Myasnikov, Ushakov and Won [MUW12].

Formally, a *power circuit* of size k is given by a pair (Γ, δ). Here, Γ is a set of k vertices and δ is a mapping $\delta : \Gamma \times \Gamma \to \{-1, 0, +1\}$. The support of δ is the subset $\Delta \subseteq \Gamma \times \Gamma$ with $(P, Q) \in \Delta \iff \delta(P, Q) \neq 0$. Thus, (Γ, Δ) is a simple directed graph with edge set Δ. We call $\delta(P, Q)$ the label of the edge (P, Q). Throughout, we require that (Γ, Δ) is acyclic. In particular, $\delta(P, P) = 0$ for all vertices P. A *marking* is a mapping $M : \Gamma \to \{-1, 0, +1\}$. We can also think of a marking as a subset of Γ where each element in M has a sign (+ or −). If $M(P) = 0$ for all $P \in \Gamma$, then we simply write $M = \emptyset$. Each node $P \in \Gamma$ is associated in a natural way with a successor marking $\Lambda_P : \Gamma \to \{-1, 0, +1\}$, $Q \mapsto \delta(P, Q)$, consisting of the target nodes of outgoing edges from P. We define the *evaluation* $\varepsilon(P)$ of a node ($\varepsilon(M)$ of a marking resp.) bottom-up in the directed acyclic graph by induction:

$$\begin{aligned} \varepsilon(\emptyset) &= 0, \\ \varepsilon(P) &= 2^{\varepsilon(\Lambda_P)} && \text{for a node } P, \\ \varepsilon(M) &= \sum_P M(P)\varepsilon(P) && \text{for a marking } M. \end{aligned}$$

Note that leaves (vertices without outgoing edges) evaluate to 1, the evaluation of a marking is a real number, and the evaluation of a node P is a positive real number. Thus, $\varepsilon(P)$ and $\varepsilon(M)$ are well-defined. We have $\varepsilon(\Lambda_P) = \log_2(\varepsilon(P))$, thus the successor marking plays the role of a logarithm. We are interested only in power circuits where all markings evaluate to integers; equivalently all nodes evaluate to some positive natural number in $2^{\mathbb{N}}$.

The *power circuit representation* of an integer sequence $m_1, \ldots, m_n$ is given by a tuple $(\Gamma, \delta; M_1, \ldots, M_n)$ where (Γ, δ) is a power circuit and $M_1, \ldots, M_n$ are markings such that $\varepsilon(M_i) = m_i$. (Hence, a single power circuit can store several different numbers; a fact is crucial in the proof of Proposition 8.12, see [DLU12].)

Example 8.1. We can represent every integer in the range $[-N, N]$ as the evaluation of some marking in a power circuit with node set $\{P_0, \ldots, P_\ell\}$ such that $\varepsilon(P_i) = 2^i$ for $0 \leq i \leq \ell$ and $\ell = \lfloor \log_2 N \rfloor$. Thus, we can convert the binary notation of an integer N into a power circuit with $\mathcal{O}(\log |N|)$ vertices and $\mathcal{O}((\log |N|) \log\log |N|)$ edges.

Example 8.2. A power circuit of size k can realize $\mathrm{tow}(k)$: Let $\Gamma = \{P_1, \ldots, P_k\}$ be the power circuit with $\delta(P_i, P_{i-1}) = 1$ and $\delta(P_i, P_j) = 0$ for $j \neq i - 1$. Then we have $\varepsilon(P_k) = \tau(k)$. We call such a power circuit a chain of k nodes.

A sum $m = \sum_{i=0}^{k} b_i 2^i$ is called *compact* if $b_i b_{i+1} = 0$ for all i. A marking M is *compact* if the sum $\sum_P M(P)\varepsilon(P)$ is compact and a power circuit is called *compact* if all its markings are compact.

Proposition 8.3 ([MUW12])**.** *Compact sums are unique. More precisely, if* $\sum_{i=0}^{k} b_i 2^i = \sum_{i=0}^{\ell} c_i 2^i$ *are compact sums, then* $k = \ell$ *and* $b_i = c_i$ *for all* i*.*

Moreover, compact sums use the least possible number of non-zero digits, i. e., if $m = \sum_{i=0}^{k} b_i 2^i$ *is a compact sum and* $m = \sum_{i=0}^{\ell} c_i 2^i$ *with* $c_i \in \{-1, 0, 1\}$*, then* $|\{i \mid b_i \neq 0\}| \leq |\{i \mid c_i \neq 0\}|$.

A power circuit Γ is called *reduced* if $\varepsilon(P) = \varepsilon(Q)$ implies $P = Q$ for $P, Q \in \Gamma$.

Proposition 8.4 ([MUW11, DLU12])**.** *The following operations can be performed in quadratic time. Input a power circuit* (Γ, δ) *of size* n *and two markings* M_1 *and* M_2*. Decide whether all markings evaluate to integers. If "yes":*

- *Compute a reduced power circuit with markings* M_i' *such that* $\varepsilon(M_i') = \varepsilon(M_i)$.
- *Decide whether* $\varepsilon(M_1) \leq \varepsilon(M_2)$.
- *Compute a new power circuit with markings* M*,* X *and* U *such that*
  - (i) $\varepsilon(M) = \varepsilon(M_1) \pm \varepsilon(M_2)$.
  - (ii) $\varepsilon(M) = 2^{\varepsilon(M_1)} \cdot \varepsilon(M_2)$.
  - (iii) $\varepsilon(M_1) = 2^{\varepsilon(X)} \cdot \varepsilon(U)$ *and either* $U = \emptyset$ *or* $\varepsilon(U)$ *is odd.*

With Comparison in Power Circuits we mean the following problem. Input: A power circuit (Γ, δ) where all vertices evaluate to natural numbers with two markings M_1 and M_2. Decide whether $\varepsilon(M_1) \leq \varepsilon(M_2)$.

Theorem 8.5. Comparison in Power Circuits *is* P*-complete.*

Proof. By Proposition 8.4, we only need to show the hardness part. We describe a LOGSPACE reduction from the Circuit Value Problem to Comparison in Power Circuits. The Circuit Value Problem is defined as follows: On input of a Boolean circuit (see Appendix A) with one output gate, together with inputs for the circuit, compute the value at the output gate of the circuit. W. l. o. g. we may assume that the circuit only consists of input and output gates, $\neg$ gates, and $\vee$ gates of fan-in two. There are no depth restrictions imposed.

More formally, we can describe the input for the Circuit Value Problem as follows: A directed acyclic graph where each vertex (in the following called *gate*) is labeled either by 0 or 1 (only vertices with no incoming edge), $\neg$ (only vertices with exactly one incoming edge), $\vee$ (only vertices with two incoming edges) or with *output* (exactly one vertex without outgoing edges and with exactly one incoming edge). We assume that all circuits are layered. More precisely, we assign a level to every gate. Input gates have level 1, and gates on level k may only receive inputs from gates on level $k - 1$.

Given such a circuit, every gate g evaluates to a truth value in a natural way, which we denote by $\mathrm{eval}(g) \in \{0, 1\}$; in particular, the task is to compute $\mathrm{eval}(\mathit{output})$. The Circuit Value Problem is well-known to be P-complete, see e. g. [Sip96, Thm. 10.44].

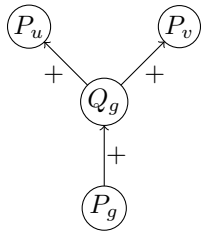


Figure 8.1.: Power circuit for $\vee$ gates.

Starting with some Boolean circuit, we design a power circuit (Γ, δ) such that, for every gate g on level k (except the output gate), there is some vertex P_g such that

$$\begin{aligned} &\varepsilon(P_g) \in 2^{\mathbb{N}}, \\ &\varepsilon(P_g) < \operatorname{tow}(2k+2)/2, && \text{and} \\ &\varepsilon(P_g) < \operatorname{tow}(2k+1)/2 && \text{if } \operatorname{eval}(g) = 0, \\ &\varepsilon(P_g) \geq \operatorname{tow}(2k+1) && \text{if } \operatorname{eval}(g) = 1. \end{aligned} \tag{8.1}$$

In the power circuit, the edges point to the "opposite direction" as in the Boolean circuit we start with. Let D be the depth of the circuit, i. e., the output gate has level $D+1$. For all $2 \leq k \leq D+1$, we create a vertex $R_k \in \Gamma$ with $\varepsilon(R_k) = \operatorname{tow}(2k)$ values and a vertex $S_k \in \Gamma$ with $\varepsilon(S_k) = \operatorname{tow}(2k-1)/2$ – this results in a power circuit of size $\mathcal{O}(D)$. Moreover, we create the following vertices and edges:

- For every input gate g with value 0, we create a single vertex $P_g \in \Gamma$ without outgoing edges. Then we have $\varepsilon(P_g) = 1 < \operatorname{tow}(3)/2$; hence, (8.1) holds in this case.
- For every input gate g with value 1, we create a chain of three vertices and define $P_g \in \Gamma$ to be the last of these three vertices, i. e., $\varepsilon(P_g) = 4 = \operatorname{tow}(3) < \operatorname{tow}(4)/2$; hence, (8.1) holds again.
- For every $\vee$ gate g with incoming edges from gates u and v, we create vertices $P_g, Q_g \in \Gamma$ and set $\delta(P_g, Q_g) = \delta(Q_g, P_u) = \delta(Q_g, P_v) = 1$.
- For every $\neg$ gate g on level k with incoming edge from gate u, we create a vertex $P_g \in \Gamma$ and set $\delta(P_g, R_k) = \delta(P_g, S_k) = 1$, and $\delta(P_g, P_u) = -1$.
- For the output gate *output* on level $D+1$ with incoming edge from gate u, we create a vertex $P_{output} \in \Gamma$ and set $\delta(P_{output}, P_u) = 1$.

It is straightforward to see that this construction can be computed in LOGSPACE: The depth can be computed by just following an arbitrary path from some input gate to the output gate. The construction of the nodes R_k, S_k can be done by increasing some counter and in every step outputting some new nodes. The construction of the

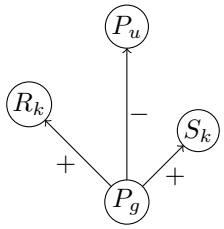


Figure 8.2.: Power circuit for ¬ gates.

nodes P_g, Q_g corresponding to gates of the circuit is also no problem as simply every gate has to be "replaced" by a constant number of vertices.

Now, we have $\varepsilon(P_{output}) \geq \varepsilon(R_{D+1}) = \mathrm{tow}(2D+2)$ if and only if $\mathrm{eval}(output) = 1$. Moreover, all nodes evaluate to numbers in $2^{\mathbb{N}}$.

In order to prove these statements, it suffices to show (8.1). We proceed by induction starting from the input gates. If g is an input gate, we have already seen that (8.1) holds. Now, let g be some $\vee$ gate on level $k \geq 2$ with incoming edges from gates u and v. If $\mathrm{eval}(u) = \mathrm{eval}(v) = 0$, then we have by induction $\varepsilon(P_u), \varepsilon(P_v) \leq \mathrm{tow}(2k-1)/2 - 1$; hence,

$$\varepsilon(P_g) = 2^{2^{\varepsilon(P_u)+\varepsilon(P_v)}} \leq 2^{2^{2\cdot(\mathrm{tow}(2k-1)/2-1)}} < 2^{\mathrm{tow}(2k)/2} \leq \mathrm{tow}(2k+1)/2.$$

With the same calculation (with $2k$ replaced by $2k+1$) we see that $\varepsilon(P_g) \leq \mathrm{tow}(2k+2)/2$ also if $\mathrm{eval}(u)$ or $\mathrm{eval}(v) = 1$. On the other hand, if $\mathrm{eval}(u) = 1$ – i. e., by induction $\varepsilon(P_u) \geq \mathrm{tow}(2k-1)$, then we have

$$\varepsilon(P_g) = 2^{2^{\varepsilon(P_u)+\varepsilon(P_v)}} > 2^{2^{\mathrm{tow}(2k-1)}} = \mathrm{tow}(2k+1).$$

In all cases, we obviously have $\varepsilon(P_g) \in 2^{\mathbb{N}}$. Now, let g be some ¬ gate on level $k \geq 2$ with incoming edge from gate u. If $\mathrm{eval}(u) = 0$, then we have by induction $\varepsilon(P_u) < \mathrm{tow}(2k-1)/2$. Hence,

$$\begin{aligned}\varepsilon(P_g) &= 2^{\mathrm{tow}(2k)+\mathrm{tow}(2k-1)/2-\varepsilon(P_u)} \\ &\geq 2^{\mathrm{tow}(2k)+\mathrm{tow}(2k-1)/2-\mathrm{tow}(2k-1)/2} = \mathrm{tow}(2k+1).\end{aligned}$$

and

$$\varepsilon(P_g) \leq 2^{\mathrm{tow}(2k)+\mathrm{tow}(2k-1)/2} \leq 2^{2\,\mathrm{tow}(2k)-1} < \mathrm{tow}(2k+2)/2.$$

If $\mathrm{eval}(u) = 1$, we have $\varepsilon(P_u) \geq \mathrm{tow}(2k-1)$ by induction; thus, as $k \geq 2$,

$$\varepsilon(P_g) = 2^{\mathrm{tow}(2k)+\mathrm{tow}(2k-1)/2-\varepsilon(P_u)} < 2^{\mathrm{tow}(2k)-1} = \mathrm{tow}(2k+1)/2,$$

and $\varepsilon(P_g) \in 2^{\mathbb{N}}$ since

$$\mathrm{tow}(2k) + \mathrm{tow}(2k-1)/2 - \varepsilon(P_u) \geq \mathrm{tow}(2k) + \mathrm{tow}(2k-1)/2 - \mathrm{tow}(2k)/2 \geq 0. \quad \square$$

8.1.1. Division in Power Circuits

The complexity of the divisibility problem in power circuits is an open question. Here, Divisibility in Power Circuits is the following problem: Given a power circuit and two markings M_1 and M_2, decide whether $\varepsilon(M_1) \mid \varepsilon(M_2)$, i. e., whether $\varepsilon(M_1)$ divides $\varepsilon(M_2)$. In the next section, divisibility will become important when dealing with the conjugacy problem in the Baumslag group.

The straightforward algorithm for Divisibility in Power Circuits transforms $\varepsilon(M_1)$ and $\varepsilon(M_2)$ first into binary and computes $\varepsilon(M_1)/\varepsilon(M_2)$ after that. The first part involves a non-elementary explosion. Still we have:

Lemma 8.6. Divisibility in Power Circuits *is decidable in non-elementary time.*

We suspect that Divisibility in Power Circuits is extremely difficult, i. e., that it cannot be solved in elementary time. However, we do not know any non-trivial lower bound. A trivial lower bound is that it is hard for P. This is immediate since Comparison in Power Circuits is P-hard. Indeed, take the two markings M_1 and M_2 of Theorem 8.5 and create two new vertices P, Q with $\Lambda_P = M_1$ and $\Lambda_Q = M_2$. Then $\varepsilon(M_1) \leq \varepsilon(M_2)$ if and only if $\varepsilon(P)$ divides $\varepsilon(Q)$.

In the following, we want to show that the standard approaches trial division and calculating modulo fail for solving Divisibility in Power Circuits in elementary time. After that, in Proposition 8.10, we will see that in a very special case, at least, modulo can be calculated in polynomial time.

Remark 8.7. In [MUW12], it is shown that in power circuits already division by 3 can lead to a non-elementary blow up: For $\ell \in \mathbb{N}$, we have

$$\frac{2^{2\ell} - 1}{3} = \sum_{i=0}^{\ell-1} 2^{2i} = S,$$

which is a compact sum; in particular, by Proposition 8.3, at least ℓ nodes are required to represent S as a power circuit. If $2^{2\ell} - 1$ is a large integer represented by a small power circuit (e. g. by a linear chain), then a power circuit to represent S is non-elementarily larger than the power circuit to represent $2^{2\ell} - 1$, see Figure 8.3.

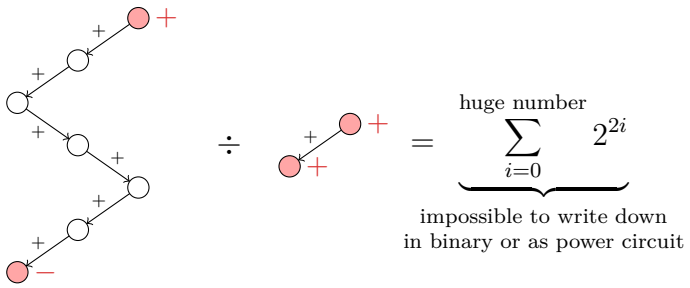
On the other hand, calculating the value of some marking modulo[1] 3 is a simple task:

Remark 8.8. Let (Γ, δ) be a power circuit with a marking M. Then a marking M' with $\varepsilon(M') = \varepsilon(M) \bmod 3$ can be computed in quadratic time: Let $L = \{P \in \Gamma \mid \varepsilon(P) = 1\}$, and $V_i = \left\{P \in \Gamma \;\middle|\; \sum_{Q \in L} \Lambda_P(Q) \equiv i \mod 2\right\}$ for $i = 0, 1$. We have

$$\varepsilon(M) \equiv \sum_{P \in V_0} M(P) + 2 \cdot \sum_{P \in V_1} M(P) \mod 3.$$

Note that if Γ is reduced this can be done even in time linear in the size of the support of M.

[1] For $a, b \in \mathbb{N}$, we define $a \bmod b$ to be the unique number $c \in \{0, \ldots, b-1\}$ with $a \equiv c \mod b$.



$$(111111\cdots11)_2 \quad \div \quad (11)_2 \quad = (10101\cdots01)_2$$

Figure 8.3.: Non-elementary blow-up when dividing by three.

With the same approach as in Remark 8.8, one can show that calculating modulo some number $m \in \mathbb{N}$ is always possible in time polynomial in m and the size of the power circuit. However, when calculating modulo a number given by a power circuit this approach might not be feasible anymore. The following theorem shows that when calculating modulo a non-elementary blow-up can occur, see also Figure 8.4.

Theorem 8.9. *Calculating modulo in power circuits can lead to a non-elementary blow-up. More precisely, for every $k \geq 4$, there is a power circuit of size $k+4$ with two markings M_1 and M_2 such that every power circuit representing $\varepsilon(M_1) \bmod \varepsilon(M_2)$ has at least $\mathrm{tow}(k)$ vertices.*

Proof. Let Γ consist of a chain of $k+2$ vertices and two additional vertices, i. e., $\Gamma = \{u_1, \ldots, u_{k+2}, v, w\}$, and

$$\Delta = \{(u_i, u_{i-1}) \mid i \in \{1, \ldots, k+2\}\} \cup \{(v, u_k), (v, u_{k-1}), (w, v)\}\,.$$

All edges are labeled with 1. We define markings M_1 and M_2 by $M_1(w) = 1$ and $M_2(u_{k+2}) = 1$, $M_2(u_{k+1}) = M_2(u_1) = -1$ and all other values are set to zero. For an example of this construction with $k = 6$, see Figure 8.4 (note that there the markings are depicted in two different power circuits for better readability). We denote $\kappa = \varepsilon(u_k) = \mathrm{tow}(k)$ and $\lambda = \varepsilon(u_{k+1}) = \mathrm{tow}(k+1)$. As we have $\varepsilon(v) = \kappa\lambda$, it follows that $\varepsilon(M_1) = 2^{\kappa\lambda}$, and also $\varepsilon(M_2) = 2^\lambda - \lambda - 1$. For $\varepsilon(M_1) \bmod \varepsilon(M_2)$, this means

$$\begin{aligned} \varepsilon(M_1) = \left(2^\lambda\right)^\kappa &\equiv (\lambda+1)^\kappa && \bmod (2^\lambda - \lambda - 1) \\ &= \sum_{i=0}^{\kappa} \binom{\kappa}{i} \lambda^i. \end{aligned}$$

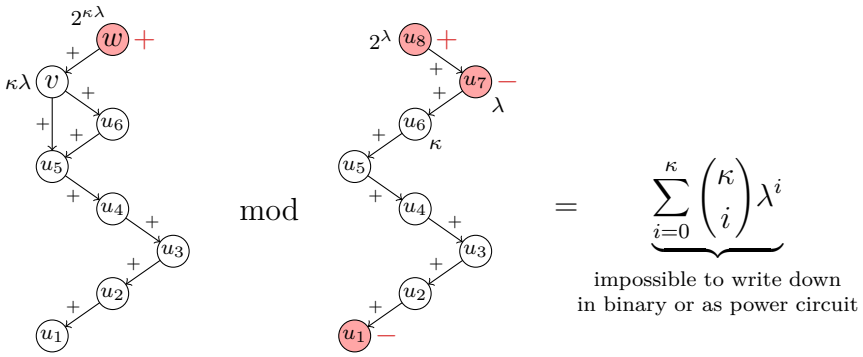
By our assumption $k \geq 4$, we have $\kappa \geq 16$, and thus $\kappa^2 + \kappa \leq 2^\kappa - 1$. Hence,

$$(\lambda+1)^\kappa \;\leq\; (2 \cdot 2^\kappa)^\kappa \;=\; 2^{\kappa^2+\kappa} \;\leq\; 2^{2^\kappa - 1} \;\leq\; 2^\lambda - \lambda - 1;$$

and thus, $\varepsilon(M_1) \bmod \varepsilon(M_2) = \sum_{i=0}^{\kappa} \binom{\kappa}{i} \lambda^i$. Moreover, $\binom{\kappa}{i} \leq \lambda/2$ for all i implies that if we represent $\binom{\kappa}{i}$ as a compact sum S_i, then the sum $\sum_{i=0}^{\kappa} S_i \lambda^i = \varepsilon(M_1) \bmod \varepsilon(M_2)$ is

compact. By Proposition 8.3, every power circuit to represent this sum needs at least κ vertices. □

We have seen that in the general case there is no way to calculate modulo without non-elementary blow-up. However, if we restrict to a very special class of power circuits, modulo can be calculated in polynomial time.



$$\left(2^\lambda\right)^\kappa \quad \text{mod} \quad 2^\lambda - \lambda - 1 \quad = \quad (\lambda+1)^\kappa$$

Figure 8.4.: Non-elementary blow-up when calculating modulo: Circuit Γ as in the proof of Proposition 8.10 with $k = 6$.

Proposition 8.10. *Let (Γ_1, δ_1) be a power circuit with a marking M_1 and (Γ_2, δ_2) a reduced power circuit with a marking M_2. Moreover, let M_2 as well as every successor marking Λ_P for $P \in \Gamma_2$ point to at most two nodes and to at most one node with value other than 1 (hence, $\varepsilon(M_2) = 2^\ell \pm 1$ or $\varepsilon(M_2) = 2^\ell$ for some $\ell \in \mathbb{N}$). Then there is a power circuit of polynomial size with a marking M such that $\varepsilon(M) = \varepsilon(M_1) \bmod \varepsilon(M_2)$, and it can be calculated in polynomial time.*

Proof. W. l. o. g. we may assume that both power circuits are reduced and that both markings have positive values. In particular, for every power of two, there is at most one vertex in Γ_1 having that value. We construct a new power circuit with a marking M with $\varepsilon(M) = \varepsilon(M_1) \bmod \varepsilon(M_2)$ by starting with $\Gamma_1 \cup \Gamma_2$ (where $\delta(P, Q) = \delta_i(P, Q)$ if both $P, Q \in \Gamma_i$ for $i = 1, 2$, and $\delta(P, Q) = 0$ for $P \in \Gamma_1$ and $Q \in \Gamma_2$ or vice-versa) and adding new vertices whenever required.

In the case that $\varepsilon(M_2)$ is a power of two (i. e., if it points to a single vertex), we can compute a marking M in the power circuit $\Gamma_1 \cup \Gamma_2$ with $\varepsilon(M) \equiv \varepsilon(M_1) \mod \varepsilon(M_2)$ as follows:

$$M(P) = \begin{cases} M_1(P) & \text{if } P \in \Gamma_1, \varepsilon(P) < \varepsilon(M_2), \\ 0 & \text{otherwise.} \end{cases}$$

By Proposition 8.4, this computation is possible in polynomial time. Because Γ_1 is reduced, we have $\varepsilon(M) \leq \sum_{i=0}^{\varepsilon(M_2)-1} 2^i < \varepsilon(M_2)$ and also $\varepsilon(M) > -\varepsilon(M_2)$. Hence, it remains to check whether $\varepsilon(M) < 0$, and if so, set $M(Q) = 1$ where $Q \in \Gamma_2$ is the unique vertex with $M_2(Q) = 1$.

Now, let $\varepsilon(M_2) = \varepsilon(Q) - (-1)^e$ for some $e \in \{0, 1\}$ and $Q \in \Gamma_2$ with $\varepsilon(Q) > 1$. We fix this vertex Q. Similar to the case before, we can calculate

$$\varepsilon(M_1) \bmod \varepsilon(M_2) = \left(\sum_{P \in \Gamma_1} M_1(P) \cdot \Big(\varepsilon(P) \bmod \varepsilon(M_2) \Big) \right) + c \cdot \varepsilon(M_2)$$

for some $c \in \mathbb{Z}$ with $|c| \leq |\Gamma_1|$. Hence, it suffices to compute a marking for $\varepsilon(P) \bmod \varepsilon(M_2)$ for every $P \in \Gamma_1$. We have

$$\varepsilon(\Lambda_P) = \varepsilon(\Lambda_Q) \cdot \lfloor \varepsilon(\Lambda_P)/\varepsilon(\Lambda_Q) \rfloor + \varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q);$$

hence, we obtain

$$\begin{aligned} 2^{\varepsilon(\Lambda_P)} &= 2^{\varepsilon(\Lambda_Q) \cdot \lfloor \varepsilon(\Lambda_P)/\varepsilon(\Lambda_Q) \rfloor + \varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q)} \\ &\equiv (-1)^{e \cdot \lfloor \varepsilon(\Lambda_P)/\varepsilon(\Lambda_Q) \rfloor} \cdot 2^{\varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q)} \mod 2^{\varepsilon(\Lambda_Q)} - (-1)^e. \end{aligned}$$

For now, assume that we already have constructed a marking with the value $\varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q)$. We can construct a vertex with value $2^{\varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q)}$ by introducing a new vertex pointing to this marking. Hence, if $e = 0$, we have obtained a marking with value $\varepsilon(P) \bmod \varepsilon(M_2)$.

Otherwise, we have to calculate $\lfloor \varepsilon(\Lambda_P)/\varepsilon(\Lambda_Q) \rfloor \bmod 2$. This can be done as follows: Because

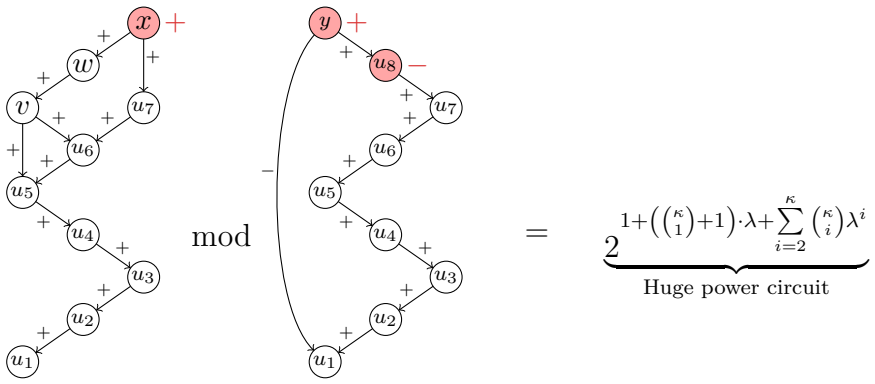
$$\lfloor \varepsilon(\Lambda_P)/\varepsilon(\Lambda_Q) \rfloor = \frac{\varepsilon(\Lambda_P) - (\varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q))}{\varepsilon(\Lambda_Q)},$$

we only need to compare the highest powers of two which divide $(\varepsilon(\Lambda_P) - (\varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q)))$ and $\varepsilon(\Lambda_Q)$ – what can be done in polynomial time by Proposition 8.4. If these coincide, then $\lfloor \varepsilon(\Lambda_P)/\varepsilon(\Lambda_Q) \rfloor \bmod 2 = 1$, otherwise $\lfloor \varepsilon(\Lambda_P)/\varepsilon(\Lambda_Q) \rfloor \bmod 2 = 0$. Now, we have

$$\begin{aligned} &2^{\varepsilon(\Lambda_P)} \bmod (2^{\varepsilon(\Lambda_Q)} + 1) \\ &\qquad = \begin{cases} 2^{\varepsilon(\Lambda_Q)} + 1 - 2^{\varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q)} & \text{if } \lfloor \varepsilon(\Lambda_P)/\varepsilon(\Lambda_Q) \rfloor \bmod 2 = 1, \\ 2^{\varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q)} & \text{if } \lfloor \varepsilon(\Lambda_P)/\varepsilon(\Lambda_Q) \rfloor \bmod 2 = 0. \end{cases} \end{aligned}$$

It remains to construct markings with value $\varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q)$ for all $P \in \Gamma_1$. As also $\varepsilon(\Lambda_Q) = 2^k \pm 1$ or $\varepsilon(\Lambda_Q) = 2^k$ for some k, this can be done by induction.

The whole algorithm proceeds as follows: for all pairs $P \in \Gamma_1$, $Q \in \Gamma_2$, a marking with value $\varepsilon(\Lambda_P) \bmod \varepsilon(\Lambda_Q)$ is computed in topologically sorted order starting with vertices which have successor markings pointing to at most one vertex. As only polynomially many markings have to be computed and each of these markings points to at most $\mathcal{O}(\Gamma_1)$ vertices, the whole algorithm runs in polynomial time and also the resulting power circuit has polynomial size. □



$$2^{2^{2\lambda\kappa}+\lambda} \quad \text{mod} \quad 2^{2^{2\lambda}-1}-2^{2\lambda} \quad = \quad 2^{2\lambda}\cdot\left(2^{2^{\kappa\lambda} \bmod (2^{2\lambda}-\lambda-1)}\right)$$

Figure 8.5.: Non-elementary blow-up when calculating modulo with all markings pointing to only two vertices.

Note that the requirement "to at most one node with value other than 1" cannot be dropped. To see this, consider the power circuit constructed in the proof of Theorem 8.9, add two new vertices x and y, and extend Δ by $\{(x,w),(x,u_{k+1}),(y,u_1),(y,u_{k+2})\}$ where (y,u_1) has label -1 and the other three edges have label 1. We create two markings M_1' and M_2' with $M_1'(x)=1$ and $M_2'(y)=1$ and $M_2'(u_{k+2})=-1$. Now, all markings, including all successor markings, point to at most two vertices, see Figure 8.5. With κ,λ being as above, we have

$$\begin{aligned}\varepsilon(M_1) \bmod \varepsilon(M_2) &= 2^{2^{\kappa\lambda}+\lambda} \bmod (2^{2^{2\lambda}-1}-2^{2\lambda}) \\ &= 2^{2\lambda}\cdot\left(2^{2^{\kappa\lambda}} \bmod (2^{2^{2\lambda}-\lambda-1}-1)\right) \\ &= 2^{2\lambda}\cdot\left(2^{2^{\kappa\lambda} \bmod (2^{2\lambda}-\lambda-1)}\right).\end{aligned}$$

In the proof of Theorem 8.9, we have seen that a reduced compact power circuit to represent $2^{\kappa\lambda} \bmod (2^{2\lambda}-\lambda-1)$ uses at least $\mathrm{tow}(k)$ vertices. From such a power circuit, a reduced compact power circuit for $2^{2\lambda}\cdot\left(2^{2^{\kappa\lambda} \bmod (2^{2\lambda}-\lambda-1)}\right)$ is obtained easily by creating a new vertex with successor marking $\left(2^{\kappa\lambda} \bmod (2^{2\lambda}-\lambda-1)\right)+\lambda$. Since this new power circuit also uses at least $\mathrm{tow}(k)$ vertices, by Proposition 8.3, every power circuit to represent $2^{2\lambda}\cdot\left(2^{2^{\kappa\lambda} \bmod (2^{2\lambda}-\lambda-1)}\right)$ needs at least $\mathrm{tow}(k)$ vertices.

8.2. Conjugacy in the Baumslag Group

The Baumslag group $\mathbf{G}_{1,2}$ (sometimes referred to as *Baumslag-Gersten group*) is an HNN extension of the Baumslag-Solitar group $\mathbf{BS}_{1,2}$. In the following, we abbreviate $\mathbf{BS}_{1,2}=\langle a,t \mid tat^{-1}=a^2\rangle=\mathbb{Z}[1/2]\rtimes\mathbb{Z}$ by H. The group H contains infinite cyclic

subgroups $A = \langle a \rangle$ and $T = \langle t \rangle$ with $A \cap T = \{1\}$. The *Baumslag group* $\mathbf{G}_{1,2}$ is the HNN extension

$$\mathbf{G}_{1,2} = \left\langle H, y \;\middle|\; yay^{-1} = t \right\rangle.$$

Note that the generator t of H is now redundant and we obtain $\mathbf{G}_{1,2}$ as a group generated by a, y with a single defining relation $yay^{-1}a = a^2yay^{-1}$, i. e., it is a one-relator group. As such it has decidable word problem [Mag32].

We denote the graph of groups which corresponds to the HNN extension $\langle H, y \mid yay^{-1} = t \rangle$ by $\mathcal{G}$; thus, the edge set of the graph is $\{y, \overline{y}\}$. As usual, we represent elements of $\mathbf{G}_{1,2}$ by $\mathcal{G}$-factorizations, i. e., as words $w = h_0 y^{\varepsilon_1} h_1 \dots y^{\varepsilon_n} h_n$ with $\varepsilon_i \in \{1, -1\}$ and $h_i \in H$. In particular, we have $|w|_{\mathcal{G}} = |w|_y + |w|_{\overline{y}}$.

If w is some $\mathcal{G}$-factorization, then at most $|w|_{\mathcal{G}}/2$ applications of Britton reductions of the type $ya^k\overline{y} \to t^k$ or $\overline{y}t^k y \to a^k$ for $k \in \mathbb{Z}$ are possible on w. However, there can be a non-elementary blow-up in the binary numbers representing the elements of H as the following example shows:

Example 8.11. Define words $w_0 = t$ and $w_{k+1} = y\, w_k\, a\, \overline{w_k}\, \overline{y}$ for $k \geq 0$. Then we have $|w_k| = 2^{k+2} - 3$ but $w_k =_{\mathbf{G}_{1,2}} t^{\mathrm{tow}(k)}$.

Britton reductions are effective because we can check whether $h = a^\ell$ (resp. $h = t^\ell$) in H (in fact, the results in Section 7.1 imply that the subgroup membership problem for A and T in H can be solved in TC^0). Thus, on input of some word $w \in \{a, \bar{a}, t, \bar{t}, y, \overline{y}\}^*$, we can effectively calculate a Britton-reduced word $\widehat{w}$ with $w =_{\mathbf{G}_{1,2}} \widehat{w}$. However, because the word $\hat{w}$ might be non-elementarily longer than w (as in Example 8.11), we need another way to represent intermediate results – this is where we can use power circuits.

Representation of Elements of $\mathbf{G}_{1,2}$. The *power circuit representation* of $h = [k, s, m-k] \in H$, where $[k, s, m-k]$ is a triple representation of h as defined in Section 7.1, is given by a power circuit and markings K, S, L such that $\varepsilon(K) = k$, $\varepsilon(S) = s$, and $\varepsilon(L) = m - k$.

The *power circuit representation* of some word $w = w_1 \cdots w_n \in (H \cup \{y, \overline{y}\})^*$ consists of a power circuit (Γ, δ), and for each $w_i = [k_i, s_i, \ell_i] \in H$, of a triple of markings $[K_i, S_i, L_i]$ with $\varepsilon(K_i) = k_i$, $\varepsilon(S_i) = s_i$, and $\varepsilon(L_i) = \ell_i$, and for each $w_i \notin H$, of a single letter y or $\overline{y}$ for $1 \leq i \leq n$. Furthermore, some technical conditions are imposed like all the markings are required to have disjoint support[2]. These conditions are necessary to make the algorithms in [DLU12] work in time $\mathcal{O}(N^3)$ – otherwise the

[2]For completeness, we list these conditions as they are stated in the proof of [DLU12, Thm. 16]:

- All the markings have disjoint support.
- For all triples $[K_i, S_i, L_i]$, the nodes in the support of K_i have no incoming edges.
- For all triples $[K_i, S_i, L_i]$, the nodes in the support of S_i have only incoming edges from nodes in the support of K_i.
- If there is an edge from the (P, Q) from a node P in the support of K_i to a node Q in the support of S_i, then $\delta(P, Q) = -S_i(Q)$.

bound is $\mathcal{O}(N^4)$, only. However, they are trivially met, when converting a word in $\{a, \bar{a}, t, \bar{t}, y, \overline{y}\}^*$ into a power circuit representation; moreover, they are not needed for the further understanding of the algorithm for the conjugacy problem – so we do not need to care about them. We define the size of a power circuit representation as $n + |\Gamma|$. Note that this might be less than the number of bits required to store the power circuit.

Now, let $w = w_1 \cdots w_N \in \{a, \bar{a}, t, \bar{t}, y, \overline{y}\}^*$ be some word. A power circuit representation of w of size $\mathcal{O}(N)$ can be constructed as follows: For every $w_i \in \{a, \bar{a}, t, \bar{t}\}$, a node P_i in Γ and a triple of markings $[K_i, S_i, L_i]$ are created such that either S_i or L_i evaluate to ± 1 (using this vertex P_i) and the other two markings are zero.

In [MUW11], the first polynomial time algorithm for the word problem of $\mathbf{G}_{1,2}$ has been given with a running time estimated by $\mathcal{O}(N^7)$ where N is the size of the input. In [DLU12], Diekert, Laun and Ushakov improved this to cubic time. Actually, any word can be Britton-reduced within this time bound:

Proposition 8.12 ([MUW11, DLU12])**.** *There is a cubic time algorithm for the following problem: given a power circuit representation of some word $w \in (H \cup \{y, \overline{y}\})^*$, compute a power circuit representation of a Britton-reduced $\mathcal{G}$-factorization $\widehat{w}$ such that $w =_{\mathbf{G}_{1,2}} \widehat{w}$. Moreover, the size for the power circuit representation of $\widehat{w}$ is linear in the size of the power circuit representation of w.*

Corollary 8.13. *There is a cubic time algorithm which on input of a word $w \in \{a, \bar{a}, t, \bar{t}, y, \overline{y}\}^*$, computes a power circuit representation of a cyclically reduced $\mathcal{G}$-factorization $\widehat{w}$ such that $w \sim_{\mathbf{G}_{1,2}} \widehat{w}$. The size for the power circuit representation of $\widehat{w}$ is linear in $|w|$.*

Proof. As remarked before, a word w of length N can be transformed into a power circuit representation of size $\mathcal{O}(N)$. Hence, we can apply Proposition 8.12 in order to obtain a power circuit representation of a Britton-reduced $\mathcal{G}$-factorization $\tilde{w}$ with $w =_{\mathbf{G}_{1,2}} \tilde{w}$. In order to reduce $\tilde{w}$ cyclically, by Lemma 3.18, we have to apply Britton reductions to $\tilde{w}\tilde{w}$ and, possibly, to a fixed cyclic permutation of $\tilde{w}$. All this can be done in cubic time by Proposition 8.12. □

Now, we are ready to deal with conjugacy. A weaker variant of the next theorem was proven by Beese [Bee12] in his Diploma thesis (supervised by the author). However, Beese could only show exponential time instead of polynomial time.

Theorem 8.14. *The following computation can be performed in time $\mathcal{O}(N^4)$. Input: words $v, w \in \{a, \bar{a}, t, \bar{t}, y, \overline{y}\}^*$ of total length N. Decide whether $|\widehat{v}|_{\mathcal{G}} > 0$ for a cyclically reduced $\mathcal{G}$-factorization $\widehat{v}$ with $\widehat{v} \sim_{\mathbf{G}_{1,2}} v$. If "yes", decide $v \sim_{\mathbf{G}_{1,2}} w$, and in the positive case, compute a power circuit representation of some $z \in \mathbf{G}_{1,2}$ such that $zvz^{-1} =_{\mathbf{G}_{1,2}} w$.*

Proof. Due to Corollary 8.13, we may assume that input words v and w are given as cyclically reduced power circuit representations. In particular, $\widehat{v} = v$ and $|\widehat{v}|_{\mathcal{G}} > 0$. Let us write $v = h_0 y^{\varepsilon_1} h_1 \ldots y^{\varepsilon_n} h_n$ as $\mathcal{G}$-factorization where $\varepsilon_i = \pm 1$. If all $\varepsilon_i = +1$, then we replace v and w by $\overline{v}$ and $\overline{w}$. Hence, without restriction there exists some $\varepsilon_i = -1$.

After a possible transposition we may assume that $v = y^{\varepsilon_1}h_1 \cdots y^{\varepsilon_n}h_n$ with $\varepsilon_1 = -1$. Since w is cyclically Britton-reduced, too, Collins' Lemma, Theorem 3.19 states: If $v \sim_{\mathbf{G}_{1,2}} w$, then $|w|_{\mathcal{G}} = n$ and after some transposition we can write $w = y^{\varepsilon_1}h_1' \cdots y^{\varepsilon_n}h_n'$ as $\mathcal{G}$-factorization. Moreover, still by Theorem 3.19, we now have

$$v \sim_{\mathbf{G}_{1,2}} w \iff \exists\, k \in \mathbb{Z} : w =_{\mathbf{G}_{1,2}} a^k v a^{-k}.$$

The key point is that k is unique and that we find an efficient way to calculate it.[3] In order to do so, we distinguish three cases:

Case $n = 1$. We have $v = \overline{y}(r,m)$ and $w = \overline{y}(s,q)$ for some $(r,m),(s,q) \in \mathbb{Z}[1/2] \rtimes \mathbb{Z}$. Now, $a^k v =_{\mathbf{G}_{1,2}} w a^k$ if and only if $(0,k)(r,m) =_H (s,q)(k,0)$. This forces $k = q - m$. Hence,

$$v \sim_{\mathbf{G}_{1,2}} w \iff 2^{q-m}r = s + 2^q(q-m) \quad \text{for } n = 1.$$

By Proposition 8.4, all required arithmetic operations can be performed in quadratic time.

Case $n \geq 2$ and $\varepsilon_2 = +1$. Then $v = \overline{y}(r,m)yh_2 \cdots y^{\varepsilon_n}h_n$ and $w = \overline{y}(s,q)yh_2' \cdots y^{\varepsilon_n}h_n'$ for some $(r,m),(s,q) \in \mathbb{Z}[1/2] \rtimes \mathbb{Z}$. We have $r \neq 0 \neq s$ since v and w are Britton-reduced. For every $k \in \mathbb{Z}$ and every Britton-reduced $\mathcal{G}$-factorization $\overline{y}\widetilde{h}_1 y \ldots y^{\varepsilon_n}\widetilde{h}_n =_{\mathbf{G}_{1,2}} a^k v a^{-k}$, we have $\widetilde{h}_1 \in t^k(r,m)T$, and hence $\widetilde{h}_1 = (2^k r, p)$ for some $p \in \mathbb{Z}$. We conclude that there is a unique $k \in \mathbb{Z}$ such that $a^k v a^{-k} =_{\mathbf{G}_{1,2}} \overline{y}\,(2^k r, p) y \cdots y^{\varepsilon_n}\widetilde{h}_n$ and $2^k r$ is an odd integer. This means we may assume from the very beginning that r and s are odd integers. Under this assumption, if $a^k v a^{-k} =_{\mathbf{G}_{1,2}} w$, then necessarily $k = 0$, and hence $v = w$ in $\mathbf{G}_{1,2}$. We obtain the following algorithm to decide $v \sim_{\mathbf{G}_{1,2}} w$:

- For $h_1 = (r,m)$ and $h_1' = (s,q)$, calculate unique $k, \ell \in \mathbb{Z}$ such that $2^k r$ and $2^\ell s$ are odd integers using Proposition 8.4.
- Decide whether $a^k v a^{-k} =_{\mathbf{G}_{1,2}} a^\ell w a^{-\ell}$ using Proposition 8.12. If "yes", then $v \sim_{\mathbf{G}_{1,2}} w$, otherwise $v \not\sim_{\mathbf{G}_{1,2}} w$.

Case $n \geq 2$ and $\varepsilon_2 = -1$. Then we have $v = \overline{y}(r,m)\overline{y}\,h_2 \cdots y^{\varepsilon_n}h_n$ and $w = \overline{y}(s,q)\overline{y}\,h_2' \cdots y^{\varepsilon_n}h_n'$. For every $k \in \mathbb{Z}$, we can write $a^k v a^{-k}$ in some Britton-reduced form which looks like $\overline{y}\,\widetilde{h}_1\overline{y} \cdots y^{\varepsilon_n}\widetilde{h}_n$. Now, $\widetilde{h}_1 \in t^k(r,m)A$. Thus, there is a unique $k \in \mathbb{Z}$ (necessarily $k = -m$) such that $\widetilde{h}_1 = (p,0)$ for some $p \in \mathbb{Z}[1/2]$. Using the same arguments as above, we obtain the following algorithm: For $h_1 = (r,m)$ and $h_1' = (s,q)$, decide whether $a^{-m} v a^m =_{\mathbf{G}_{1,2}} a^{-q} w a^q$. If "yes", then $v \sim_{\mathbf{G}_{1,2}} w$, otherwise $v \not\sim_{\mathbf{G}_{1,2}} w$.

By Proposition 8.12, the tests $a^k v a^{-k} =_{\mathbf{G}_{1,2}} w$ can be performed in cubic time. All other computations can be done in quadratic time by Proposition 8.4. Since all transpositions of the $\mathcal{G}$-factorization for w have to be considered, this yields an $\mathcal{O}(N^4)$-algorithm. □

[3] Beese calculates in [Bee12] this value k and computes certain normal forms which are checked for equivalence. This leads to an exponential time algorithm.

Now, we can apply the results of Chapter 4 to our algorithm for the conjugacy problem in the Baumslag group.

Theorem 8.15. *Let Σ be some finite monoid generating set of $\mathbf{G}_{1,2}$. There is an algorithm that decides doubly strongly generically in time $\mathcal{O}(N^4)$ on input of two words $v, w \in \Sigma^*$ with $|vw| \leq N$ whether $v \sim_{\mathbf{G}_{1,2}} w$. Moreover, if Σ is symmetric and inputs are restricted to cyclically freely reduced words, the algorithm is still doubly strongly generically in time $\mathcal{O}(N^4)$.*

Proof. If Σ is not the standard alphabet $\{a, \bar{a}, t, \bar{t}, y, \bar{y}\}^*$, then the words v, w can be rewritten as words over $\{a, \bar{a}, t, \bar{t}, y, \bar{y}\}$ in linear time and with only a linear increase of length. Now, by Theorem 8.14, there is an algorithm for solving the conjugacy problem in time $\mathcal{O}(N^4)$ if one of the input words cannot be conjugated into H; by Theorem 4.21, these words form a strongly generic set. □

For the remainder of the section, the situation is as follows: We have $v = (r, m) \in \mathbb{Z}[1/2] \rtimes \mathbb{Z}$ and $w = (s, q) \in \mathbb{Z}[1/2] \rtimes \mathbb{Z}$, both can be assumed to be in power circuit representation. We may assume $v \neq 1 \neq w$ in $\mathbf{G}_{1,2}$. After a conjugation with some t^k where k is large enough, we may assume that $r, s \in \mathbb{Z}$. If $m = 0$, we replace v by $yv\bar{y}$. Hence, $m \neq 0$ and likewise $q \neq 0$, too. Recall Equation (7.2) from Section 7.1:

$$\begin{aligned}(r, m) \sim_H (s, q) \iff \\ m = q \text{ and } \exists\, \ell \in \{0, \ldots, m-1\} : r \cdot 2^\ell - s \equiv 0 \mod (2^m - 1).\end{aligned} \tag{8.2}$$

By Lemma 8.6, we are able to perform the test whether $(r, m) \sim_H (0, m)$ and $(s, q) \sim_H (0, q)$ in non-elementary time. Assume that one of the answers is "no". Say, $(r, m) \not\sim_H (0, m)$. Then there is no $h \in A \cup T \subseteq H$ such that $(r, m) \sim_H h$. Hence, we are in case (ii) of Theorem 3.19 and we have:

Proposition 8.16. *Let $r, m, s, q \in \mathbb{Z}$ with $m \neq 0$. If $(r, m) \not\sim_H (0, m)$, then*

$$(r, m) \sim_{\mathbf{G}_{1,2}} (s, q) \iff (r, m) \sim_H (s, q).$$

By Proposition 8.16, we may assume $(r, m) \sim_H (0, m)$, $(s, q) \sim_H (0, q)$, and $(r, m) \not\sim_H (s, q)$, i. e., we are in case (i) of Theorem 3.19. To verify this involves perhaps non-elementary procedures. However, it remains to decide $(0, m) \sim_{\mathbf{G}_{1,2}} (0, q)$, only. The last test is polynomial time again, even for power circuits.

Proposition 8.17. *Let $m, q \in \mathbb{Z}$. Then we have*

$$(0, m) \sim_{\mathbf{G}_{1,2}} (0, q) \iff (m, 0) \sim_H (q, 0) \iff \exists\, k \in \mathbb{Z} : m = 2^k q.$$

Proof. The assertion $(m, 0) \sim_H (q, 0) \iff \exists\, k \in \mathbb{Z} : m = 2^k q$ is clear since $(m, 0) = a^m$ and $(q, 0) = a^q$ in $H = \mathbf{BS}_{1,2}$. Let $(0, m) \sim_{\mathbf{G}_{1,2}} (0, q)$. We have to show $(m, 0) \sim_H (q, 0)$ since the other direction is trivial. We have $(q, 0) \sim_{\mathbf{G}_{1,2}} (0, q)$. Let $z = h_0 y^{\varepsilon_1} h_1 \cdots y^{\varepsilon_n} h_n$ be a Britton-reduced $\mathcal{G}$-factorization such that $z^{-1}(q, 0)z =_{\mathbf{G}_{1,2}}$

$(0,m)$. Since ${h_0}^{-1}(q,0)h_0 = (p,0)$ for some $p \neq 0$, we have $n \geq 1$ and $\varepsilon_1 = -1$ because there has to occur a Britton reduction. Thus, $y{h_0}^{-1}(q,0)h_0\overline{y} =_{\mathbf{G}_{1,2}} t^p$. Now, ${h_1}^{-1}(0,p)h_1 \in A \cup T$ if and only if ${h_1}^{-1}(0,p)h_1 =_{\mathbf{G}_{1,2}} (0,p)$. Thus, we may assume $h_1 = 1$. Since z is Britton-reduced, we cannot have $\varepsilon_2 = +1$. Moreover, we cannot have $\varepsilon_2 = -1$ because then $y(0,p)\overline{y}$ is Britton-reduced. Thus, we must have $n = 1$ and we may choose $z = h\overline{y}$ for some $h \in H$. This means $z^{-1}(q,0)z = yh^{-1}(q,0)h\overline{y} = (0,m)$, which in turn implies $(m,0) \sim_H (q,0)$. □

Putting together Theorem 8.14, Proposition 8.16, and Proposition 8.17, we obtain:

Corollary 8.18. *The following problem is decidable in non-elementary time. Input: power circuit representations v, w for elements of $\mathbf{G}_{1,2}$. Question: $v \sim_{\mathbf{G}_{1,2}} w$?*

Remark 8.19. Let us highlight that integer division can be reduced to the conjugacy problem in $\mathbf{BS}_{1,2}$. For $m \geq 1$, we obtain as a special case of (8.2) and a well-known fact from elementary number theory

$$(0,m) \sim_{\mathbf{BS}_{1,2}} (2^s - 1, m) \iff 2^m - 1 \mid 2^s - 1 \iff m \mid s. \tag{8.3}$$

If we allow a power circuit representation for integers, then this reduction from Divisibility in Power Circuits to conjugacy can be computed in polynomial time. Hence, as no elementary time algorithm for Divisibility in Power Circuits is known, also no elementary algorithm is known to solve the conjugacy problem in $\mathbf{BS}_{1,2}$ for inputs in power circuit representation.

Corollary 8.20. *If there is no elementary-time algorithm to solve* Divisibility in Power Circuits *then the conjugacy problem in the Baumslag group $\mathbf{G}_{1,2}$ is non-elementary in the average case even for a unary representation of group elements.*

Because of Corollary 8.20, we suspect that also the conjugacy problem in $\mathbf{G}_{1,2}$ is not solvable in elementary time.

Proof. Assume that the conjugacy problem in the Baumslag group $\mathbf{G}_{1,2}$ is elementary on the average, i. e., there is an elementary function $f : \mathbb{N} \to \mathbb{N}$ and an algorithm which decides the conjugacy problem in $\mathbf{G}_{1,2}$ in time $f(N)$ on average. We give an elementary-time algorithm to solve Divisibility in Power Circuits. Let (Γ, δ) be a power circuit of size k with markings M and S such that $\varepsilon(M) = m$ and $\varepsilon(S) = s$. For each node $P \in \Gamma$, it is easy to construct a word $u_P \in \{a, \overline{a}, t, \overline{t}, y, \overline{y}\}^*$ such that $u_P =_{\mathbf{G}_{1,2}} t^{\varepsilon(P)}$ and $|u_P| \leq k^k$ following the scheme from Example 8.11. Hence, in time $2^{\mathcal{O}(k \log k)}$ we can construct words $v, w \in \{a, \overline{a}, t, \overline{t}, y, \overline{y}\}^*$ of length $2^{\mathcal{O}(k \log k)}$ such that $v =_{\mathbf{G}_{1,2}} (0,m)$ and $w =_{\mathbf{G}_{1,2}} (2^s - 1, m)$. By Remark 8.19 and Proposition 8.16, we have $m \mid s$ if and only if $v \sim_{\mathbf{G}_{1,2}} w$. Hence, the algorithm for the conjugacy problem in $\mathbf{G}_{1,2}$ can be applied. As the number of words of length $2^{\mathcal{O}(k \log k)}$ is at most $2^{2^{\mathcal{O}(k \log k)}}$, its running time is bounded by

$$2^{2^{\mathcal{O}(k \log k)}} \cdot f\left(2^{\mathcal{O}(k \log k)}\right). \qquad \square$$

Computer Experiments. Theorem 8.15 states that the conjugacy problem in $\mathbf{G}_{1,2}$ is strongly generically in P. However, it does not give an exact quantification of the size of the strongly generic set where the algorithm runs in polynomial time. Lemma B.5 provides us some bound, but it might not be tight. Therefore, we have conducted computer experiments: We sampled $11 \cdot 10^9$ (i. e., 11 billion) random words $w \in \{a, \bar{a}, t, \bar{t}, y, \bar{y}\}^*$ with $4 \leq |w|_{\mathcal{G}} = 2n \leq 24$, see Figure 8.6, in order to estimate the probability $\Pr_n[w \in H]$.

The experiments confirm $\Pr_n[w \in H] \approx 0$. Moreover, for $n = 14$, our random process did not find a single $w \in H$. The initial values seem to suggest $\Pr_n[w \in H] \in \mathcal{O}(0.25^n)$. This is much better than the upper bound of Lemma B.5 with $|\{a, \bar{a}, t, \bar{t}, y, \bar{y}\}| = 6$ – which is no surprise as we used very rough estimations in Lemma B.4, only.

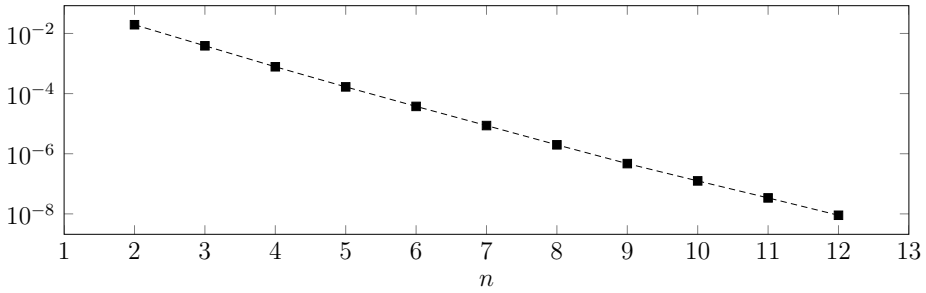


Figure 8.6.: Portion of reduced words $x \in H$ with $|x|_{\mathcal{G}} = 2n$, sampling $11 \cdot 10^9$ words.

Chapter 9.

Conclusion and Open Questions

To conclude, we provide a brief overview of the thesis in which we also address some conjectures and open questions for further research.

We have examined the complexity of conjugacy in various HNN extensions and amalgamated products (or more generally, fundamental groups of graphs of groups). Moreover, we have seen that in most cases the elements which cannot be conjugated into any vertex group form a strongly generic set. Thus, if conjugacy can be decided in case (iii) of the Conjugacy Criterion, Theorem 3.19, then it is doubly strongly generically decidable. By applying this result, we have established that the conjugacy problem of the Baumslag group is doubly strongly generically in P. Moreover, the conjugacy problem of a fundamental group of a finite graph of groups is doubly strongly generically in P if either all vertex groups are free abelian and there are at least two edges in the graph of groups, or all edge groups are finite and the word problem of all vertex groups is in P (note that we did not use any statement about the conjugacy problem in the vertex groups).

We have also improved some undecidability results concerning HNN extensions of free and free abelian groups. These results leave us the following open questions:

(1) Is there an HNN extension of $\mathbb{Z}^2$ with undecidable conjugacy problem? Note that by [BMV10, Cor. 6.14], in the case of surjective inclusions of the associated subgroups, we have decidability.

(2) What is the minimal number of stable letters for an HNN extension of $\mathbb{Z}^3$ to have undecidable conjugacy problem?

(3) What is the minimal rank of the associated subgroups for an HNN extension (with one stable letter) of a non-abelian free group to have undecidable conjugacy problem?

If we restrict to one stable letter, we could show that conjugacy in HNN extensions of free abelian groups is decidable (Proposition 5.18).

(4) We conjecture that the conjugacy problem of an HNN extension of a free abelian group with one stable letter is in P.

Concerning fundamental groups of finite graphs of groups with f. g. free or free abelian vertex groups, also the following questions are open:

(5) Are the word and conjugacy problem in fundamental groups of finite graphs of groups with f. g. free or free abelian vertex groups and cyclic edge groups in NC or even in LOGSPACE? In Remark 5.11, we state that they are both in P; for cyclic vertex groups, we know that they are in LOGDCFL by Theorem 7.23.

(6) Is the word problem in fundamental groups of finite graphs of groups with f. g. free groups in P? We conjecture that this is true and that it can be shown using compression techniques as in [Loh06, LS07, Sch08].

(7) Is the conjugacy problem in fundamental groups of finite graphs of groups with f. g. free groups strongly generically decidable? This is true for Miller's group [BMR07c]. Also, if we replace "strongly generically" by "generically", it is true due to [KMSS03, Thm. C] (see Theorem 2.7).

Let $\mathcal{G}$ be a finite graph of groups with finite edge groups. In Chapter 6, we have shown that under some reasonable assumptions for T, conjugacy in $\pi_1(\mathcal{G})$ can be decided in time $\mathcal{O}(\log N \cdot T(N))$ on a RAM if in all vertex groups it can be decided in time $T(N)$ (N is the length of the input).

(8) Is it true that under the same assumptions, the conjugacy problem in $\pi_1(\mathcal{G})$ is decidable in time $\mathcal{O}(T(N))$?

In the same situation we have shown that the word (resp. conjugacy) problem in $\pi_1(\mathcal{G})$ can be solved by a NC^2 circuit with oracle gates for the word (resp. conjugacy) problem of the vertex groups – in some special cases, in fact, it is NC^1-Turing-reducible to the word (resp. conjugacy) problem in the vertex groups.

(9) Is it in general true that

$$\mathrm{WP}(\pi_1(\mathcal{G})) \in \mathsf{NC}^1(\{\mathrm{WP}(G_P) \mid P \in V(Y)\} \cup \{\mathrm{WP}(F_2)\}) \qquad \text{and}$$
$$\mathrm{CP}(\pi_1(\mathcal{G})) \in \mathsf{NC}^1(\{\mathrm{CP}(G_P) \mid P \in V(Y)\} \cup \{\mathrm{WP}(F_2)\})\,?$$

In particular, if G_1, G_2 have word problem in LOGSPACE and $C \leq G_1, G_2$ is finite, is the word problem of $G_1 *_C G_2$ in LOGSPACE?
Or does a counter-example exist (given that $\mathsf{NC}^1 \neq \mathsf{NC}^2$)?

The word and conjugacy problem of generalized Baumslag-Solitar groups (fundamental groups of finite graphs of groups with infinite cyclic vertex and edge groups) are decidable in LOGDCFL. For linear GBS groups we know that the word problem is in LOGSPACE. This asks for a generalization:

(10) Is the word problem of all GBS groups in LOGSPACE?

A positive solution would imply that also the conjugacy problem is in LOGSPACE.

Finally, we treated conjugacy in the Baumslag group and its relation to division in power circuits. For Divisibility in Power Circuits we could show that the straightforward algorithms cannot work in elementary time.

(11) Which lower bounds can be proven for Divisibility in Power Circuits? Can Divisibility in Power Circuits be decided in elementary time? We conjecture that the answer is "no". This would also imply that the conjugacy problem in $\mathbf{G}_{1,2}$ cannot be solved in elementary time.

(12) Based on strong evidence, we conjecture that the word problem of the Baumslag group $\mathbf{G}_{1,2}$ is in NC, although Comparison in Power Circuits is hard for P. This discrepancy is due to the observation that we have a logarithmic bound on the length of paths in power circuits which are obtained in the solution of the word problem of $\mathbf{G}_{1,2}$.

Appendix A.

Basics of Complexity

This chapter is *not* part of the thesis. It summarizes some basic notions of complexity theory and fixes our model of random access machines.

The classes of the Landau notation for functions from $\mathbb{N}$ to the non-negative reals $\mathbb{R}_{\geq 0}$ are defined as follows

$$\begin{aligned}
\mathcal{O}(f) &= \left\{ g : \mathbb{N} \to \mathbb{R} \;\middle|\; \limsup_{N\to\infty} \frac{g(N)}{f(N)} < \infty \right\}, \\
o(f) &= \left\{ g : \mathbb{N} \to \mathbb{R} \;\middle|\; \lim_{N\to\infty} \frac{g(N)}{f(N)} = 0 \right\}, \\
\Omega(f) &= \{ g : \mathbb{N} \to \mathbb{R} \mid f \in \mathcal{O}(g) \}, \qquad \text{and} \\
\Theta(f) &= \mathcal{O}(f) \cap \Omega(f).
\end{aligned}$$

LOGSPACE Computable Functions and Reductions. A function $f : \Sigma^* \to \Delta^*$ is said to be computable in LOGSPACE if there is a deterministic Turing machine with read-only input tape and write-only output tape, which computes $f(w)$ using only $\mathcal{O}(\log(|w|))$ space on the work tape. A formal definition of a Turing machine can be found in Section 5.1.1 or in some textbook like [Sip96].

Let $K \subseteq \Sigma^*$ and $L \subseteq \Delta^*$. A function $f : \Sigma^* \to \Delta^*$ is a (many-to-one) LOGSPACE *reduction* from K to L if it is computable in LOGSPACE and $w \in K$ if and only if $f(w) \in L$. It is a standard fact that LOGSPACE reducibility is transitive; also, if K is LOGSPACE-reducible to L and $L \in$ LOGSPACE, then $K \in$ LOGSPACE, too.

Oracle Turing Machines. We also need another type of reduction: A Turing machine with *oracle* for L has an additional oracle tape, which is write-only, and three additional special states $\{q_{\text{Query}}, q_{\text{Yes}}, q_{\text{No}}\}$. If the Turing machine enters the q_{Query} state, in the next step the oracle tape is deleted and the machine goes into the state q_{Yes} or q_{No} depending on whether the word on the oracle tape is in L or not. A Turing machine can also have more than one oracle. Let $\mathcal{C}$ be some complexity class defined by Turing machines and let L be some arbitrary language. Then a language K is in $\mathcal{C}^L$ if there is a $\mathcal{C}$-Turing machine with oracle for L which accepts K. If $\mathcal{D}$ is some arbitrary complexity class, then $\mathcal{C}^{\mathcal{D}}$ denotes the class of languages accepted by some $\mathcal{C}$ Turing machine with oracles for some finite set $\{L_1, \ldots, L_k\} \subseteq \mathcal{D}$. It is a standard fact that $\mathsf{LOGSPACE}^{\mathsf{LOGSPACE}} = \mathsf{LOGSPACE}$ and $\mathsf{P}^{\mathsf{P}} = \mathsf{P}$.

For some complexity class $\mathcal{C}$ we define $\mathsf{LOGSPACE}^{\mathcal{C}}$-computable functions and $\mathsf{LOGSPACE}^{\mathcal{C}}$ reductions the same way as $\mathsf{LOGSPACE}$-computable functions and $\mathsf{LOGSPACE}$ reductions.

Random Access Machines For a finer measure of complexity within P, we use random access machines. Our model of randoms access machines is essentially the model presented in [Pap94, Sec. 2.6]. A *random access machine (RAM)* consist of a finite sequence of commands (i. e., a program or algorithm) and a (infinite) set of registers $(X_i)_{i\in\mathbb{Z}}$, each of them storing an integer. Access to the registers is either by direct (read/write x_i for some $i \in \mathbb{Z}$) or indirect addressing (read/write x_{x_i} for some $i \in \mathbb{Z}$). The following operations can be performed in one time step (i. e., we are applying the uniform cost model):

- arithmetic operations: addition, subtraction, division by two,
- conditional jumps to a fixed position in the program code: conditions allowing comparisons $<, =, >$ and logical connectives $\wedge, \vee, \neg$.

In particular, we do not allow arbitrary multiplication of two registers. For our purposes, we do not need multiplication as an intrinsic instruction; moreover, omitting it guarantees a good bound for the running time when transforming the RAM into a Turing machine. The following observations are easy to see:

- Everything computable by a Turing machine in time $T(N)$ is computable by a RAM in time $\mathcal{O}(T(N))$.
- If some algorithms runs in time $T(N)$ on a RAM, then the number of bits required to store the registers is in $\mathcal{O}(T(N))$.
- There is a polynomial p such that if some algorithm runs in time $T(N)$ on a RAM, then it runs in time $p(T(N))$ on a Turing machine.

In particular, some algorithm runs in polynomial time on a RAM if and only if it runs in polynomial time on a Turing machine – so, for the definition of P, both Turing machines or RAMs might be used.

Circuit Complexity A (*Boolean*) *circuit* is a directed acyclic graph where each vertex corresponds to some gate and each edge to some connection. There are *input gates* with no incoming edges, *output gates* with one incoming edge and no outgoing edges, and inner gates like Boolean gates (and $\wedge$, or $\vee$, not $\neg$) or MAJORITY gates. The number of incoming edges of an inner gate is called its *fan-in*. The *size* of a circuit is its number of gates. The *depth* is the length of the longest path from some input gate to some output gate. Every circuit computes a Boolean function from the input gates to the output gates in a natural way. The outcome of a MAJORITY gate is 1 if at least half of the inputs are 1, otherwise it is 0.

Let $i \in \mathbb{N}$. A formal language L is said to be decidable in NC^i (resp. a function f computable in NC^i) if there is a family of circuits (one circuit for each input length) using Boolean gates of fan-in 2 (resp. 1 for $\neg$) of depth $\mathcal{O}((\log N)^i)$ and size polynomial in N where N is the length of the input. Similarly AC^i and TC^i are defined as the class of languages decided by families of circuits of depth $\mathcal{O}((\log N)^i)$ and size polynomial in N with the following gates:

- AC^i uses $\wedge$ and $\vee$ gates of unbounded fan-in and $\neg$ gates.
- TC^i uses $\wedge$, $\vee$ and MAJORITY gates of unbounded fan-in and $\neg$ gates.

Likewise, the classes of AC^i- and TC^i-computable functions are defined. Since with this definition all the classes contain undecidable problems, some uniformity conditions are imposed. A family of circuits is said to be LOGSPACE *uniform* if there is a LOGSPACE Turing machine which on input a^N computes a proper encoding of the circuit in the family which has N input gates. A family of circuits is said to be DLOGTIME *uniform* if there is a DLOGTIME Turing machine which accepts the direct connection language of the circuit family (i. e., on input (a^N, i, j, k) it decides whether in the N-input-circuit gate number k is the j-th input of gate number i and also computes the type of gate number i). We call the classes AC^0, TC^0, and NC^1 *uniform* if they are DLOGTIME uniform. These uniform classes are all contained in LOGSPACE. All other of the above classes are called *uniform* if they are LOGSPACE uniform. They all contain LOGSPACE. In the following, we only consider uniform classes without emphasizing it further. For a more formal and precise definition of circuits and uniformity conditions (including DLOGTIME Turing machines), we refer to the textbook [Vol99].

Appendix B.

More on Amenability

In this chapter, we give additional proofs for the results of Chapter 4: Section B.1 presents the proof of Theorem 4.3 following [CSGdlH99] and [Soa94]; in Section B.2 we give another, self-contained proof of Theorem 4.18.

B.1. Proof of Theorem 4.3

This section is devoted to the proof of Theorem 4.3. We need a well-known result from literature, also called the marriage theorem. For a proof see e. g. [Hal98, Thm. 5.1.2]. Let Γ be a graph. A *matching* is a subset $E' \subseteq E(\Gamma)$ of undirected edges such that every vertex is incident to at most one undirected edge in E'. A bipartite graph is a graph Γ such that $V(\Gamma)$ is a disjoint union $A \dot\cup B$ and there are no edges having both endpoints in A or both endpoints in B, i. e., every edge is incident to one vertex in A and to one in B.

Proposition B.1 (Hall)**.** *Let Γ be a locally finite bipartite graph with $V(\Gamma) = A \dot\cup B$. If $|\mathcal{N}(U) \setminus U| \geq |U|$ for all finite subsets $U \subseteq A$, then there exists a matching $E' \subseteq E(\Gamma)$ such that all vertices in A are incident to some edge in E'.*

Let us recall the statement of Theorem 4.3:

Theorem B.2 ([CSGdlH99, Soa94, Kes59a, Kes59b, Ger88, Gro93])**.** *Let Γ be a d-regular graph. Then the following statements are equivalent:*

(i) *There exists a map $f : V(\Gamma) \to V(\Gamma)$ such that $\sup_{v \in V(\Gamma)} d(f(v), v) < \infty$ and $|f^{-1}(v)| \geq 2$ for all $v \in V(\Gamma)$ (*Gromov condition*).*

(ii) *There exists some $k \in \mathbb{N}$ such that for every finite $U \subseteq V(\Gamma)$ we have*
$$\left|\mathcal{N}^k(U)\right| \geq 2\,|U|\,.$$

(iii) *Γ satisfies a strong isoperimetric inequality.*

(iv) *$\rho(\Gamma) < 1$.*

(v) *There is some $\sigma \in (0,1)$ such that $p^{(n)}(u,v) \in o(\sigma^n)$ for all $u, v \in V(\Gamma)$.*

If Γ is connected, also the following condition is equivalent:

(vi) *There is some $\sigma \in (0,1)$ and $u \in V(\Gamma)$ such that $p^{(n)}(u,u) \in o(\sigma^n)$.*

Proof. Here, we present the proofs from [CSGdlH99] and [Soa94]. We write $V = V(\Gamma)$ and $E = E(\Gamma)$.

(i) ⇒ (ii). We define $k = \sup_{v\in V} d(f(v), v) < \infty$ where f is as in (i). Then, $f^{-1}(U) \subseteq \mathcal{N}^k(U)$ for every finite $U \subseteq V$. Hence, $\left|\mathcal{N}^k(U)\right| \geq |f^{-1}(U)| \geq 2\,|U|$ by the definition of f.

(ii) ⇒ (i). Consider the bipartite graph Γ' with $V(\Gamma') = V \times \{1, 2, 3\}$, where $A = V \times \{1, 2\}$ consists of two disjoint copies of $V(\Gamma)$ and $B = V \times \{3\}$ consists of one copy of $V(\Gamma)$. For $i = 1, 2$, we draw an undirected edge connecting (u, i) with $(v, 3)$ if and only if $d(u, v) \leq k$, where d is the distance in Γ and k is as in (ii). By (ii), we have $\left|\mathcal{N}^k(U) \setminus U\right| \geq 2\,|U|$ for every finite set $U \subseteq V \times \{1\}$ or $U \subseteq V \times \{2\}$. Therefore, it follows that $\left|\mathcal{N}^k(U) \setminus U\right| \geq |U|$ for every finite subset $U \subseteq A$. Thus, the assumptions of Proposition B.1 are satisfied; and hence, there exists a matching E' such that every vertex in A is incident to some edge. Now, define $f : V \to V$ by $f(v) = u$ if there is an edge in E' connecting $(v, 3)$ with some $(u, 1)$ or $(u, 2)$ in A. If there is no such edge, we set $f(v) = v$. As E' is a matching, this is a well-defined function. By the definition of Γ', we have $\sup_{v\in V} d(v, f(v)) \leq k < \infty$ and $|f^{-1}(u)| \geq 2$ for all $u \in V$.

(ii) ⇔ (iii). By Lemma 4.1.

(iii) ⇒ (iv). For $x \in \mathbb{R}^V$, we define the 1-norm as $\|x\|_1 = \sum_{v\in V} |x(v)|$. Let ε be as in (iii). As a first step, we show that

$$\sum_{e\in E} |x(\iota(e)) - x(\tau(e))| \geq \varepsilon \|x\|_1 \tag{B.1}$$

for all finitely supported $x \in \ell^2(V)$ with $x(v) \geq 0$ for all $v \in V$.

By (iii), we know that $|\beta U| \geq \varepsilon\,|U|$ for all finite $U \subseteq V$. By viewing U as a function in $\ell^2(V)$ (i. e., $U = \sum_{u\in U} u$), we obtain

$$\begin{aligned}\sum_{e\in E} |U(\iota(e)) - U(\tau(e))| &= 2\left|\left\{e \in E \;\middle|\; \iota(e) \in U, \tau(e) \in \overline{U}\right\}\right| \\ &\geq 2\max\left\{|\beta U \cap U|\,, \left|\beta U \cap \overline{U}\right|\right\} \geq |\beta U| \geq \varepsilon \cdot |U| = \varepsilon \cdot \|U\|_1\,.\end{aligned}$$

Now, let $x = \sum_{i=1}^n c_i U_i$ with U_i finite, $U_i \subseteq U_{i+1}$, and $c_i > 0$. Since for fixed e the terms $U_i(\iota(e)) - U_i(\tau(e))$ and $U_j(\iota(e)) - U_j(\tau(e))$ are either both ≥ 0 or both ≤ 0, it follows that

$$\begin{aligned}\sum_{e\in E} \left|\sum_{i=1}^n c_i\Big(U_i(\iota(e)) - U_i(\tau(e))\Big)\right| &= \sum_{e\in E}\sum_{i=1}^n c_i\,|U_i(\iota(e)) - U_i(\tau(e))| \\ &= \sum_{i=1}^n c_i \sum_{e\in E} |U_i(\iota(e)) - U_i(\tau(e))| \\ &\geq \sum_{i=1}^n c_i \varepsilon\,\|U_i\|_1 = \varepsilon\,\|x\|_1\,.\end{aligned}$$

Hence, we have shown (B.1). Now, consider some arbitrary finitely supported $x \in \ell^2(V)$. We are going to apply (B.1) to x^2:

$$\begin{aligned}\varepsilon \|x\|_2^2 = \varepsilon \left\|x^2\right\|_1 &\leq \sum_{e\in E} \left|x^2(\iota(e)) - x^2(\tau(e))\right| \\ &= \sum_{e\in E} |x(\iota(e)) - x(\tau(e))| \cdot |x(\iota(e)) + x(\tau(e))| \\ &\leq \left(\sum_{e\in E} |x(\iota(e)) - x(\tau(e))|^2\right)^{\frac{1}{2}} \cdot \left(\sum_{e\in E} |x(\iota(e)) + x(\tau(e))|^2\right)^{\frac{1}{2}}.\end{aligned}$$

In the last step we used the Cauchy-Schwarz inequality. Since $\mathbb{R} \to \mathbb{R}$, $t \mapsto t^2$ is a convex function it follows that

$$\begin{aligned}|x(\iota(e)) + x(\tau(e))|^2 + |x(\iota(\bar{e})) + x(\tau(\bar{e}))|^2 &= 2\,|x(\iota(e)) + x(\tau(e))|^2 \\ &\leq |2x(\iota(e))|^2 + |2x(\tau(e))|^2;\end{aligned}$$

thus, we obtain

$$\sum_{e\in E} |x(\iota(e)) + x(\tau(e))|^2 \leq \sum_{e\in E} |2x(\iota(e))|^2 = 4d\,\|x\|_2^2,$$

and hence the so-called *Sobolev inequality*

$$\frac{\varepsilon^2}{4d}\,\|x\|_2^2 \leq \sum_{e\in E} |x(\iota(e)) - x(\tau(e))|^2$$

follows. Moreover, we have

$$\begin{aligned}\langle (1-A)x,\, x\rangle &= \sum_{v\in V} ((1-A)x)(v) \cdot x(v) \\ &= \sum_{v\in V} \frac{1}{d} \sum_{\substack{e\in V \\ \tau(e)=v}} \Big(x(\tau(e)) - x(\iota(e))\Big) \cdot x(\tau(e)) \\ &= \frac{1}{2d}\left(\sum_{e\in E}\Big(x(\tau(e)) - x(\iota(e))\Big) \cdot x(\tau(e)) + \sum_{e\in E}\Big(x(\tau(\bar{e})) - x(\iota(\bar{e}))\Big) \cdot x(\tau(\bar{e}))\right) \\ &= \frac{1}{2d} \sum_{e\in E}\Big(x(\tau(e)) - x(\iota(e))\Big)^2 \quad \geq 0.\end{aligned}$$

Finally, this leads to the desired result:

$$\begin{aligned}\rho(A) = \sup_{\substack{\|x\|_2=1, \\ |\mathrm{supp}(x)|<\infty}} |\langle Ax,\, x\rangle| &= \sup_{\substack{\|x\|_2=1, \\ |\mathrm{supp}(x)|<\infty}} |1 - \langle (1-A)x,\, x\rangle| \\ &= 1 - \inf_{\substack{\|x\|_2=1, \\ |\mathrm{supp}(x)|<\infty}} |\langle (1-A)x,\, x\rangle| \leq 1 - \frac{\varepsilon^2}{8d^2}.\end{aligned}$$

(iv) ⇒ (iii). Let $U \subseteq V$ be finite. Then we have $((1-A)U)(v) = 0$ for $v \in V \setminus \beta U$, and $|((1-A)U)(v)| \leq 1$ for $v \in \beta U$. Hence, we obtain

$$\begin{aligned} |\beta U| \geq |\langle (1-A)U,\, U\rangle| &\geq |\langle U,\, U\rangle| - |\langle AU,\, U\rangle| \\ &\geq (1-\rho(A)) \cdot \langle U,\, U\rangle \quad = (1-\rho(A))\,|U|\,. \end{aligned}$$

(iv) ⇒ (v). We have $p^{(n)}(u,v) = (A^n u)(v) \leq \|(A^n u)\| \leq \|A\|^n$, and $\|A\| = \rho(A) < 1$ by Lemma 4.2.

(v) ⇒ (vi). trivial.

(vi) ⇒ (v). Let $v, v' \in V$. Since Γ is connected, there are constants $k, \ell \in \mathbb{N}$ such that $p^{(k)}(u,v) > 0$ and $p^{(\ell)}(v',u) > 0$. Hence, $p^{(k)}(u,v) \cdot p^{(n)}(v,v') \cdot p^{(\ell)}(v',u) \leq p^{(k+n+\ell)}(u,u)$ implies that

$$p^{(n)}(v,v') \leq \frac{p^{(k+n+\ell)}(u,u)}{p^{(k)}(u,v) \cdot p^{(\ell)}(v',u)} \in o(\sigma^n).$$

(v) ⇒ (iv). First we prove the following claim:
Let $B : \ell^2(V) \to \ell^2(V)$ be some self-adjoint linear operator and $x \in \ell^2(V)$ with $\|x\|_2 = 1$. Then

$$\Big(\langle B^2 x,\, x\rangle\Big)^{2^k} \leq \langle B^{2^{k+1}} x,\, x\rangle \quad \text{for all } k \geq 0. \tag{B.2}$$

Both sides of this inequality are positive since B is self-adjoint. For $k = 0$, the statement is clear. Hence, let $k > 0$. The claim follows by induction and Cauchy-Schwarz inequality:

$$\Big(\langle B^2 x,\, x\rangle\Big)^{2^k} \leq \Big(\langle B^{2^k} x,\, x\rangle\Big)^2 \leq \langle B^{2^k} x,\, B^{2^k} x\rangle \cdot \langle x,\, x\rangle = \langle B^{2^{k+1}} x,\, x\rangle.$$

Now, fix some finitely supported $x \in \ell^2(V)$ with $\|x\|_2 = 1$. By (B.2) and the assumption (v), we have

$$\begin{aligned} \Big(\langle A^2 x,\, x\rangle\Big)^{2^k} &\leq \langle A^{2^{k+1}} x,\, x\rangle \\ &= \sum_{u,v \in V(\Gamma)} p^{(2^{k+1})}(u,v) x(u) x(v) \\ &\leq |\mathrm{supp}(x)|^2 \cdot \max_u |x(u)|^2 \cdot \max_{u,v \in \mathrm{supp}(x)} p^{(2^{k+1})}(u,v) \\ &< \sigma^{2^{k+1}} \qquad \text{for } k \text{ large enough.} \end{aligned}$$

Hence, $\langle A^2 x,\, x\rangle < \sigma^2$ and

$$\rho(A)^2 = \|A\|^2 = \sup_{\substack{\|x\|_2 = 1 \\ |\mathrm{supp}(x)| < \infty}} |\langle Ax,\, Ax\rangle| = \sup_{\substack{\|x\|_2 = 1 \\ |\mathrm{supp}(x)| < \infty}} \Big|\langle A^2 x,\, x\rangle\Big| \leq \sigma^2.$$

□

B.2. Back-to-base Probability in HNN Extensions: Another Proof of Theorem 4.18

In this section, we want to present an alternative proof of Theorem 4.18, which examines random walks in the Cayley graphs. In comparison to the first proof, it is rather lengthy and complicated. Therefore, it is not included in the main part of this thesis. Nevertheless, in order to derive the results about random walks, it does not require Theorem 4.3 – thus, it is self-contained. Hence, it might give a better insight why the words representing group elements outside the base group form a strongly generic set. This proof has been published in the preprint [DMW13].

We need some preliminary definitions: Let us define a preorder between functions from $\mathbb{N}$ to $\mathbb{R}_{\geq 0}$ as follows: We write $f \preceq g$ if there exist $k \in \mathbb{N}$ and $\varepsilon > 0$ such that for all but finitely many $n \in \mathbb{N}$ we have

$$f(n) \leq n^k g(n) + 2^{-\varepsilon n}.$$

Moreover, we write $f \approx g$ if both, $f \preceq g$ and $g \preceq f$. We are mainly interested in functions $f \approx 0$. Note that if $f \approx 0$ and $g(n) \in f(\theta(n))$, then also $g \approx 0$. The notion $f \approx g$ is, therefore, rather flexible and simplifies some formulae.

Again, we investigate the general situation of an HNN extension G which is given as $G = \langle H, t \mid tat^{-1} = \varphi(a) \text{ for } a \in A \rangle$ with a finitely generated base group H. We denote the corresponding graph of groups by $\mathcal{G}$. As before, we set $\Sigma = \Sigma' \cup \{t, \bar{t}\}$ where $\Sigma' \subseteq H$ is some finite monoid generating set of H. This defines a monoid presentation $\eta : \Sigma^* \to G$. Note that for $x \in \Sigma^*$ we have $|x|_{\mathcal{G}} = |x|_t + |x|_{\bar{t}}$. Let $\widehat{x} \in \Sigma^*$ denote a Britton-reduced $\mathcal{G}$-factorization such that $\eta(x) = \eta(\widehat{x})$ in G. Using this notation we define $\|x\|_{\mathcal{G}}$ by $\|x\|_{\mathcal{G}} = |\widehat{x}|_{\mathcal{G}}$. Observe that by Lemma 3.13 this is well defined.

With *back-to-base probability* we mean the probability that a random walk in the Schreier graph $\Gamma(G, H, \Sigma)$ ends in the base group H. For each $n \in \mathbb{N}$, we view Σ^n as a probability space with a uniform distribution. With $\Pr[\eta(x) \in H]$, we denote the probability that some x which is drawn from Σ^n uniformly at random represents some group element inside H. We have $p^{(n)}(H, H) = \Pr[\eta(x) \in H] = \sum_{g \in H} p'^{(n)}(1, g)$ where p describes the simple random walk in the Schreier graph and p' describes the simple random walk in the Cayley graph of G. Thus, we consider random walks in the Cayley graph of G w. r. t. the generating set Σ where each outgoing edge is chosen with equal probability.

Theorem B.3. *Let $G = \langle H, t \mid tat^{-1} = \varphi(a) \text{ for } a \in A \rangle$ be an HNN extension and $B = \varphi(A)$. Then $\Pr[\eta(x) \in H] \approx 0$ (as a function in n) if and only if $[H : A] \geq 2$ and $[H : B] \geq 2$.*

By Theorem 4.3, we know that Theorem B.3 is equivalent to Theorem 4.18. The proof of Theorem B.3 covers the rest of this section.

First, we consider $A = H = B$. Then G is a semidirect product $G = H \rtimes \mathbb{Z}$. Let $\psi : G \to \mathbb{Z}$ be the projection onto the second component. We have $\eta(x) \in H$ if and

only if $\psi(\eta(x)) = 0$. Since Σ can be viewed as a constant, it is not hard to see that we have $\Pr[\eta(x) \in H] \in \Theta(1/\sqrt{n})$. (Actually, if $|\Sigma|$ is not viewed as a constant we obtain a more precise estimation. Since the expected value for $|x|_{\mathcal{G}}$ is $2n/|\Sigma|$, one can show $\Pr[\eta(x) \in H] \in \Theta(\sqrt{|\Sigma|/n})$.)

The second case is $A = H \neq B$. For example, G is a solvable Baumslag-Solitar group $\mathbf{BS}_{1,q}$. We content ourselves with a lower bound on $\Pr[\eta(x) \in H]$. We begin by proving a conditional probability:

$$\Pr\Big[\eta(x) \in H \;\Big|\; |x|_{\mathcal{G}} = 2m\Big] \geq \frac{\binom{2m}{m}}{(m+1)2^{2m}} \in \Theta(m^{-1.5}).$$

To see this, observe that, due to $A = H$, a Britton reduction on a word $x \in \Sigma^*$ leads always to H if $|x|_t = |x|_{\bar{t}}$ and for every prefix y of x we have $|y|_t \geq |y|_{\bar{t}}$. Thus, we have $\eta(x) \in H$ as soon as the projection of x onto $\{t, \bar{t}\}^*$ is a Dyck word (a Dyck word is a word which describes a correct bracketing when considering t as an opening and $\bar{t}$ as a closing bracket). The number of Dyck words of length $2m$ is given by the m-th Catalan number $\frac{1}{m+1}\binom{2m}{m} \in \Theta(m^{-1.5} \cdot 4^m)$. We obtain a estimation

$$\begin{aligned}\Pr[\eta(x) \in H] &= \sum_{m=0}^{\lfloor n/2 \rfloor} \Pr\Big[|x|_{\mathcal{G}} = 2m\Big] \cdot \Pr\Big[\eta(x) \in H \;\Big|\; |x|_{\mathcal{G}} = 2m\Big] \\ &\in \Omega\left(\sum_{m=0}^{\lfloor n/2 \rfloor} \Pr\Big[|x|_{\mathcal{G}} = 2m\Big] \cdot m^{-1.5}\right) = \Omega(n^{-1.5}).\end{aligned}$$

Using Chernoff bounds[1] and the fact that the expected value for $|x|_{\mathcal{G}}$ is $2n/|\Sigma|$, we can state for $A = H$ a more precise upper and lower bound as follows:

$$\Pr[\eta(x) \in H] \in \mathcal{O}\Big(\sqrt{|\Sigma|/n}\Big) \cap \Omega\Big((|\Sigma|/n)^{1.5}\Big).$$

Finally, let us consider the most interesting case $A \neq H \neq B$. In order to finish the proof of Theorem B.3, we have to show $\Pr[\eta(x) \in H] \approx 0$.

We change probability space. We embed Σ^n into the space Σ^* with a measure $\mu_{0,n}$ on Σ^* which concentrates its mass on Σ^n (i. e., $\mu_{0,n}(\Sigma^n) = 1$) with corresponding probability $\Pr_{0,n}[\cdots]$. We now have to show that $\Pr_{0,n}[\eta(x) \in H] \approx 0$ if $A \neq H \neq B$. Let μ_m be the measure on Σ^* which is defined by reading letters from Σ each with equal probability as long as the random walk contains at most m letters from $\{t, \bar{t}\}$. More formally,

$$\mu_m(\{x\}) = \begin{cases} \frac{2}{|\Sigma|^{|x|+1}} & \text{if } |x|_{\mathcal{G}} = m, \\ 0 & \text{otherwise,} \end{cases}$$

[1] Chernoff bounds state that $\sum_{k=\lceil qn \rceil}^{n} \binom{n}{k} p^k (1-p)^{n-k} \in 2^{-\Omega(n)}$ for $0 \leq p < q \leq 1$.

for $x \in \Sigma^*$. This gives a corresponding probability on Σ^* which is concentrated on those words with $|x|_{\mathcal{G}} = m$. We denote it by $\Pr_m[\cdots]$. Still, there is a close connection between these probabilities. In particular, we have

$$\Pr_{0,n}\left[|x|_{\mathcal{G}} = m\right] = \frac{\left|\left\{x \in \Sigma^n \;\middle|\; |x|_{\mathcal{G}} = m\right\}\right|}{|\Sigma^n|} = \binom{n}{m} \cdot \frac{2^m \cdot (|\Sigma|-2)^{n-m}}{|\Sigma|^n},$$

$$\begin{aligned}\Pr_m[|x| = n] &= \left|\left\{x \in \Sigma^n \;\middle|\; |x|_{\mathcal{G}} = m\right\}\right| \cdot \Pr_m[x = w] \\ &= \binom{n}{m} \cdot 2^m \cdot (|\Sigma|-2)^{n-m} \cdot \frac{2}{|\Sigma|^{n+1}},\end{aligned}$$

where $w \in \Sigma^n$ is some fixed word with $|w|_{\mathcal{G}} = m$. Hence,

$$\begin{aligned}\Pr_{0,n}\left[|x|_{\mathcal{G}} = m\right] &= \frac{|\Sigma|}{2} \cdot \Pr_m[|x| = n], \\ \Pr_{0,n}\left[\eta(x) \in H \;\middle|\; |x|_{\mathcal{G}} = m\right] &= \Pr_m[\eta(x) \in H \mid |x| = n].\end{aligned}$$

Since $\Pr_{0,n}\left[|x|_{\mathcal{G}} = m\right] \approx 0$ for $m \leq n/|\Sigma|$, we obtain

$$\begin{aligned}\Pr_{0,n}[\eta(x) \in H] &= \sum_{m=0}^{n} \Pr_{0,n}\left[\eta(x) \in H \wedge |x|_{\mathcal{G}} = m\right] \\ &\approx \sum_{m=\lceil n/|\Sigma|\rceil}^{n} \Pr_{0,n}\left[\eta(x) \in H \wedge |x|_{\mathcal{G}} = m\right] \\ &= \sum_{m=\lceil n/|\Sigma|\rceil}^{n} \Pr_{0,n}\left[\eta(x) \;\middle|\; |x|_{\mathcal{G}} = m\right] \cdot \Pr_{0,n}\left[|x|_{\mathcal{G}} = m\right] \\ &= \sum_{m=\lceil n/|\Sigma|\rceil}^{n} \frac{|\Sigma|}{2} \cdot \Pr_m[\eta(x) \in H \mid |x| = n] \cdot \Pr_m[|x| = n] \\ &= \sum_{m=\lceil n/|\Sigma|\rceil}^{n} \frac{|\Sigma|}{2} \cdot \Pr_m[\eta(x) \in H \wedge |x| = n] \\ &\leq \sum_{m=\lceil n/|\Sigma|\rceil}^{n} \frac{|\Sigma|}{2} \cdot \Pr_m[\eta(x) \in H].\end{aligned}$$

Therefore, it is enough to show that $\Pr_m[\eta(x) \in H] \approx 0$ as a function in m.

There is also a natural probability distribution on Σ'^* which is formally defined by μ_0 (be aware that μ_0 is different from $\mu_{0,n}$). Indeed, we have $\mu_0(\Sigma'^*) = 1$ and the distribution on Σ'^* is given by a random walk which stops with probability $2/|\Sigma|$, and if it does not stop, then it chooses the next letter with equal probability. In order to emphasize that the mass of μ_0 is on Σ'^*, we also write $\Pr_{\Sigma'}[y] = \Pr_0[y]$ for $y \in \Sigma'^*$.

Lemma B.4. *For all $g \in \Sigma'^*$ and $\beta \in \{t, \bar{t}\}$, we have*

$$\Pr_{\Sigma'}\left[\eta(\beta g\, y\, \overline{\beta}) \notin H\right] \geq \frac{2}{|\Sigma|^2}.$$

Proof. By symmetry, we may assume $\beta = t$. We have to show that $\Pr_{\Sigma'}[\eta(gy) \notin A] \geq 2/|\Sigma|^2$. We consider the cases $\eta(g) \notin A$ and $\eta(g) \in A$ separately. For $\eta(g) \notin A$, we obtain

$$\Pr_{\Sigma'}[\eta(gy) \notin A] \geq \Pr_{\Sigma'}[y = 1] = 2/|\Sigma| \geq 2/|\Sigma|^2.$$

For $\eta(g) \in A$ and $a \in \Sigma'$, we have $\eta(ga) \notin A$ if and only if $\eta(a) \notin A$. Since $A \neq H$ and Σ' generates H, there must be some letter $a \in \Sigma'$ with $\eta(a) \notin A$. Therefore, in the second case

$$\Pr_{\Sigma'}[\eta(gy) \notin A] \geq \Pr_{\Sigma'}[y = a] = 2/|\Sigma|^2.$$

□

As before, a $\mathcal{G}$-factorization of $x \in \Sigma^*$ with $|x|_{\mathcal{G}} = m$ is written as a word $x = g_0\beta_1 g_1 \dots \beta_m g_m$ such that $\beta_i \in \{t, \bar{t}\}$ and $g_i \in \Sigma'^*$ for $1 \leq i \leq m$. We define random variables $X_i : \Sigma^* \to \mathbb{N}$ by $X_i(x) = \|g_0\beta_1 g_1 \dots \beta_i g_i\|_{\mathcal{G}}$ for all $0 \leq i \leq m$. Another way to explain $X_i(x)$ is as follows. Choose any prefix z of x such that $|z|_{\mathcal{G}} = i$, compute the Britton reduction $\widehat{z}$ of z and set $X_i(x) = |\widehat{z}|_{\mathcal{G}}$. The differences $Y_i = X_i - X_{i-1}$ define random variables Y_i with values in $\{-1, 1\}$. Clearly, $X_i = \sum_{j=1}^{i} Y_j$ for all $0 \leq i \leq m$. Note that $X_0 = 0$ and $X_1 = Y_1 = 1$ are constant functions.

Consider a $\mathcal{G}$-factorization $x = g_0\beta_1 g_1 \dots \beta_m g_m$ for x (as before $\beta_i \in \{t, \bar{t}\}$ for all i). For $1 \leq i \leq m$, let $\widehat{z}_{i-1}$ be Britton-reduced such that $\eta(\widehat{z}_{i-1}) = \eta(g_0\beta_1 \dots g_{i-2}\beta_{i-1})$. Then the $\mathcal{G}$-factorization of $\widehat{z}_{i-1}$ is $g'_0\beta'_1 g'_1 \dots \beta'_j g'_j$ for some $j \leq i-1$. Note that the last factor g'_j can be, a priori, any word in Σ'^*. Now, it depends only on the factors $\beta'_j g'_j$ and $g_{i-1}\beta_i$ whether or not the $\mathcal{G}$-length of the Britton-reduced word increases or decreases when reading the next factor $g_{i-1}\beta_i$ (i. e., when passing from $\widehat{z}_{i-1}$ to $\widehat{z}_i$). The probability for that is described by the random variable Y_i. For all $(\varepsilon_1, \dots, \varepsilon_{i-1}) \in \{-1, 1\}^{i-1}$, Lemma B.4 shows

$$\begin{aligned} \Pr[Y_i = 1 \mid Y_j = \varepsilon_j \text{ for } j < i] &\geq 1/2 + 1/2 \cdot 2/|\Sigma|^2 \\ &= 1/2 + 1/|\Sigma|^2. \end{aligned} \tag{B.3}$$

Let $\{Z_i \mid i = 1, \dots, m\}$ be a set of m independent random variables taking values in $\{-1, 1\}$ such that $\Pr[Z_i = 1] = 1/2 + 1/|\Sigma|^2$ for all $1 \leq i \leq m$. By (B.3), it follows that for every $1 \leq i \leq m$ and every $(\varepsilon_1, \dots, \varepsilon_{i-1}) \in \{-1, 1\}^{i-1}$ we have

$$\Pr_m[Y_i = -1 \mid Y_j = \varepsilon_j \; \forall j < i] \leq \Pr[Z_i = -1]. \tag{B.4}$$

This observation is crucial in the proof of the next lemma, which concludes the proof of Theorem B.3 because, in particular, it implies $\Pr_m[X_m = 0] \approx 0$.

Lemma B.5. *We have*

$$\Pr_m[X_m = 0] \leq \left(1 - \frac{4}{|\Sigma|^4}\right)^{m/2}.$$

Proof. The assertion is trivial for $m = 0$ or m odd. Hence, let $m \geq 2$ be even. First, let us show that for all $p \in \mathbb{Z}$, $1 \leq k \leq m$, and $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_{k-1}) \in \{-1, 1\}^{k-1}$ we have

$$\Pr_m \left[\sum_{i=k}^{m} Y_i \leq p \;\middle|\; Y_j = \varepsilon_j \; \forall j < k \right] \leq \Pr_m \left[\sum_{i=k}^{m} Z_i \leq p \right]. \tag{B.5}$$

We prove (B.5) by induction on $m - k$. The case $m = k$ is the statement of (B.4); hence, let $k < m$:

$$\begin{aligned}
&\Pr_m \left[\sum_{i=k}^{m} Y_i \leq p \;\middle|\; Y_j = \varepsilon_j \; \forall j < k \right] \\
&\quad = \sum_{\varepsilon_k = \pm 1} \Pr_m \left[Y_k = \varepsilon_k \mid Y_j = \varepsilon_j \; \forall j < k \right] \cdot \Pr_m \left[\sum_{i=k+1}^{m} Y_i \leq p - \varepsilon_k \;\middle|\; Y_j = \varepsilon_j \; \forall j \leq k \right] \\
&\quad \leq \sum_{\varepsilon_k = \pm 1} \Pr_m \left[Y_k = \varepsilon_k \mid Y_j = \varepsilon_j \; \forall j < k \right] \cdot \Pr_m \left[\sum_{i=k+1}^{m} Z_i \leq p - \varepsilon_k \right] \\
&\quad \leq \sum_{\varepsilon_k = \pm 1} \Pr_m \left[Z_k = \varepsilon_k \right] \cdot \Pr_m \left[\sum_{i=k+1}^{m} Z_i \leq p - \varepsilon_k \right] = \Pr_m \left[\sum_{i=k}^{m} Z_i \leq p \right].
\end{aligned}$$

We have to explain the inequality leading to the last line above. By (B.4), there is some $\delta_{\varepsilon,k} \geq 0$ such that $\Pr_m \left[Y_k = -1 \mid Y_j = \varepsilon_j \; \forall j < k \right] + \delta_{\varepsilon,k} = \Pr_m \left[Z_k = -1 \right]$. Thus, by definition, $\Pr_m \left[Y_k = 1 \mid Y_j = \varepsilon_j \; \forall j < k \right] - \delta_{\varepsilon,k} = \Pr_m \left[Z_k = 1 \right]$. Hence, the inequality follows from $\Pr_m \left[\sum_{i=k+1}^{m} Z_i \leq p - 1 \right] \leq \Pr_m \left[\sum_{i=k+1}^{m} Z_i \leq p + 1 \right]$.

As a special case of (B.5) for $k = 1$, we obtain

$$\Pr_m \left[X_m \leq p \right] = \Pr_m \left[\sum_{i=1}^{m} Y_i \leq p \right] \leq \Pr_m \left[\sum_{i=1}^{m} Z_i \leq p \right].$$

In order to prove the lemma, it is enough to consider $p = 0$. We conclude

$$\begin{aligned}
\Pr_m \left[X_m = 0 \right] \leq \Pr \left[\sum_{i=1}^{m} Z_i \leq 0 \right] &= \sum_{\substack{(\varepsilon_1, \ldots, \varepsilon_m) \in \{-1,1\}^m \\ |\{j \mid \varepsilon_j = 1\}| \leq m/2}} \prod_{i=1}^{m} \Pr \left[Z_i = \varepsilon_i \right] \\
&\leq 2^m \cdot \left(\frac{1}{2} - \frac{1}{|\Sigma|^2} \right)^{m/2} \cdot \left(\frac{1}{2} + \frac{1}{|\Sigma|^2} \right)^{m/2} = \left(1 - \frac{4}{|\Sigma|^4} \right)^{m/2}.
\end{aligned}$$

□

Acknowledgements

First of all I want to thank my advisor Volker Diekert who awakened my interest in algorithmic group theory, offered me the position at FMI, guided my research into the right ways, and encouraged me constantly. I am also grateful to Markus Lohrey for co-examining this thesis. Furthermore, I want to mention all my colleagues at FMI who made this time a time I will recall with pleasure: Ulrich Hertrampf, Jonathan Kausch – my officemate with whom I had so many helpful discussions –, Manfred Kufleitner, Jürn Laun, Alexander Lauser, Heike Photien, Horst Prote, Martin Seybold, Jan-Philipp Wächter, Tobias Walter. Thanks for your encouragement, your helpful comments, and the time we spent together eating cake and playing darts and tabletop soccer.

Also, thanks to all the colleagues, friends, and family members who read parts of my thesis and gave their valuable comments to improve the presentation: Jochen Briem, Camilo Carrillo, Jonathan Kausch, Catherine Lee, Alejandra Salnikov, Vitalij Salnikov, Martin Seybold, Lukas Smirek, Paolo Di Tella, Tobias Walter, Irmgard Weiß, Ortrun Weiß, Regine Weiß, Ulrich Weiß. In particular, thanks to my parents Regine and Ulrich also for supporting me during this time of thesis writing and all my life until now.

Bibliography

[AM83] Michael Anshel and Kenneth McAloon. Reducibilities among decision problems for HNN groups, vector addition systems and subsystems of Peano arithmetic. *Proc. Amer. Math. Soc.*, 89(3):425–429, 1983.

[AM84] J. Avenhaus and K. Madlener. On the complexity of intersection and conjugacy problems in free groups. *Theoret. Comput. Sci.*, 32(3):279–295, 1984.

[Ans76a] Michael Anshel. The conjugacy problem for HNN groups and the word problem for commutative semigroups. *Proc. Amer. Math. Soc.*, 61(2):223–224, 1976.

[Ans76b] Michael Anshel. Conjugate powers in HNN groups. *Proc. Amer. Math. Soc.*, 54:19–23, 1976.

[Ans76c] Michael Anshel. Decision problems for HNN groups and vector addition systems. *Math. Comput.*, 30(133):154–156, 1976.

[AS74] Michael Anshel and Peter Stebe. The solvability of the conjugacy problem for certain HNN groups. *Bull. Amer. Math. Soc.*, 80:266–270, 1974.

[Bar99] Laurent Bartholdi. Counting paths in graphs. *Enseign. Math. (2)*, 45(1-2):83–131, 1999.

[Bau69] Gilbert Baumslag. A non-cyclic one-relator group all of whose finite quotients are cyclic. *J. Austr. Math. Soc.*, 10(3-4):497–498, 1969.

[Bee11a] Benjamin Beeker. Multiple conjugacy problem in graphs of free abelian groups. *CoRR*, abs/1106.3978, 2011.

[Bee11b] Benjamin Beeker. *Problèmes géométriques et algorithmiques dans des graphes de groupes*. PhD thesis, Université de Caen Basse-Normandie, 2011.

[Bee12] Janis Beese. Das Konjugationsproblem in der Baumslag-Gersten-Gruppe. Diploma thesis, Fakultät Mathematik, Universität Stuttgart, 2012. In German.

[Bir67] A.P. Biryukov. Some algorithmic problems for finitely defined commutative semigroups. *Siberian Mathematical Journal*, 8(3):384–391, 1967.

[BK94] L. A. Bokut′ and G. P. Kukin. *Algorithmic and combinatorial algebra*, volume 255 of *Mathematics and its Applications*. Kluwer Academic Publishers Group, Dordrecht, 1994.

[BL81] A. M. Ballantyne and D. S. Lankford. New decision algorithms for finitely presented commutative semigroups. *Comput. and Maths. with Appls.*, 7:159–165, 1981.

[BMMV06] O. Bogopolski, A. Martino, O. Maslakova, and E. Ventura. Free-by-cyclic groups have solvable conjugacy problem. *B. Lond. Math. Soc.*, 38:787–794, 2006.

[BMR99] Gilbert Baumslag, Alexei Myasnikov, and Vladimir Remeslennikov. Malnormality is decidable in free groups. *Internat. J. Algebra Comput.*, 9(6):687–692, 1999.

[BMR07a] A. V. Borovik, A. G. Miasnikov, and V. N. Remeslennikov. The conjugacy problem in HNN-extensions I: regular elements, black holes and generic complexity. *Vestnik OMGU*, Special Issue:103–110, 2007.

[BMR07b] Alexandre V. Borovik, Alexei G. Myasnikov, and Vladimir N. Remeslennikov. The conjugacy problem in amalgamated products. I. Regular elements and black holes. *Internat. J. Algebra Comput.*, 17(7):1299–1333, 2007.

[BMR07c] Alexandre V. Borovik, Alexei G. Myasnikov, and Vladimir N. Remeslennikov. Generic complexity of the conjugacy problem in HNN-extensions and algorithmic stratification of Miller's groups. *Internat. J. Algebra Comput.*, 17(5-6):963–997, 2007.

[BMV10] O. Bogopolski, A. Martino, and E. Ventura. Orbit decidability and the conjugacy problem for some extensions of groups. *Trans. Amer. Math. Soc.*, 362(4):2003–2036, 2010.

[BO93] Ron Book and Friedrich Otto. *String-Rewriting Systems*. Springer-Verlag, 1993.

[Boo59] W. W. Boone. The Word Problem. *Annals of Mathematics*, 70(2):207–265, 1959.

[Bri63] John L. Britton. The word problem. *Ann. of Math.*, 77:16–32, 1963.

[BS62] Gilbert Baumslag and Donald Solitar. Some two-generator one-relator non-Hopfian groups. *Bull. Amer. Math. Soc.*, 68:199–201, 1962.

[Car75] Edward W. Cardoza. Computational complexity of the word problem for commutative semigroups. Master thesis, MIT, 1975.

[CJ12] Matthew J. Craven and Henri C. Jimbo. Evolutionary algorithm solution of the multiple conjugacy search problem in groups, and its applications to cryptography. *Groups Complexity Cryptology*, 4:135–165, 2012.

[CK14] Bren Cavallo and Delaram Kahrobaei. A polynomial time algorithm for the conjugacy problem in $\mathbb{Z}^n \rtimes \mathbb{Z}$. *Reports@SCM*, 1(1):55–60, 2014.

[CLM76] E. Cardoza, R. Lipton, and A. R. Meyer. Exponential space complete problems for Petri nets and commutative semigroups: preliminary report. In *Eighth Annual ACM Symposium on Theory of Computing (Hershey, Pa., 1976)*, pages 50–54. Assoc. Comput. Mach., New York, 1976.

[CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 3 edition, 2009.

[Coh82] Joel M. Cohen. Cogrowth and amenability of discrete groups. *J. Funct. Anal.*, 48(3):301–309, 1982.

[Col69] Donald J. Collins. On embedding groups and the conjugacy problem. *J. London Math. Soc. (2)*, 1:674–682, 1969.

[CSGdlH99] T. Ceccherini-Silberstein, R. I. Grigorchuk, and P. de la Harpe. Amenability and paradoxical decompositions for pseudogroups and discrete metric spaces. *Tr. Mat. Inst. Steklova*, 224:68–111, 1999.

[DDM10] Volker Diekert, Andrew J. Duncan, and Alexei G. Myasnikov. Geodesic rewriting systems and pregroups. In O. Bogopolski, I. Bumagin, O. Kharlampovich, and E. Ventura, editors, *Combinatorial and Geometric Group Theory*, Trends in Mathematics, pages 55–91. Birkhäuser, 2010.

[DDM12] Volker Diekert, Andrew J. Duncan, and Alexei G. Myasnikov. Cyclic rewriting and conjugacy problems. *Groups Complexity Cryptology*, 4(2):321–355, 2012.

[Deh11] Max Dehn. Über unendliche diskontinuierliche Gruppen. *Math. Ann.*, 71(1):116–144, 1911.

[Dic13] Leonard Eugene Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *American Journal of Mathematics*, 35(4):413–422, 1913.

[Dic80] Warren Dicks. *Groups, Trees and Projective Modules*. Lecture Notes in Mathematics. Springer, 1980.

[DK14] Volker Diekert and Jonathan Kausch. Logspace computations in graph products. In Katsusuke Nabeshima, Kosaku Nagasaka, Franz Winkler, and Ágnes Szántó, editors, *ISSAC*, pages 138–145. ACM, 2014.

[DKL12] Volker Diekert, Jonathan Kausch, and Markus Lohrey. Logspace computations in graph groups and Coxeter groups. In D. Fernãndez-Baca, editor, *Proceedings of LATIN 2012*, volume 7256 of *Lecture Notes in Computer Science*, pages 243–254. Springer, 2012.

[DKR13] Volker Diekert, Manfred Kufleitner, and Gerhard Rosenberger. *Elemente der Diskreten Mathematik*. Walter de Gruyter, 2013.

[DLU12] Volker Diekert, Jürn Laun, and Alexander Ushakov. Efficient algorithms for highly compressed data: The word problem in Higman's group is in P. *International Journal of Algebra and Computation*, 22(8):1–19, 2012.

[DMW13] Volker Diekert, Alexei Myasnikov, and Armin Weiß. Conjugacy in Baumslag's group, generic case complexity, and division in power circuits. *CoRR*, abs/1309.5314, 2013.

[DMW14] Volker Diekert, Alexei G. Myasnikov, and Armin Weiß. Conjugacy in Baumslag's Group, Generic Case Complexity, and Division in Power Circuits. In Alberto Pardo and Alfredo Viola, editors, *LATIN*, volume 8392 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2014.

[DW13] Volker Diekert and Armin Weiß. Context-Free Groups and Bass-Serre Theory. *ArXiv e-prints*, 2013.

[EEO13] Murray Elder, Gillian Elston, and Gretchen Ostheimer. On groups that have normal forms computable in logspace. *J. Algebra*, 381:260–281, 2013.

[Eme58] Vladimir A. Emelichev. Commutative semigroups with one defining relation. *Shuya Gosudarstvennyi Pedagogicheskii Institut Uchenye Zapiski*, 6:227–242, 1958.

[FMR68] Patrick C. Fischer, A. R. Meyer, and Arnold L. Rosenberg. Counter machines and counter languages. *Math. Systems Theory*, 2:265–283, 1968.

[FMR10] Elizaveta Frenkel, Alexei G. Myasnikov, and Vladimir N. Remeslennikov. Regular sets and counting in free groups. In *Combinatorial and geometric group theory*, Trends Math., pages 93–118. Birkhäuser/Springer Basel AG, Basel, 2010.

[Føl55] Erling Følner. On groups with full Banach mean value. *Math. Scand.*, 3:243–254, 1955.

[Fru77] M. A. Frumkin. Polynomial time algorithms in the theory of linear Diophantine equations. In *Fundamentals of computation theory (Proc. Internat. Conf., Poznań-Kórnik, 1977)*, volume 56 of *Lecture Notes in Computer Science*, pages 386–392. Springer, Berlin, 1977.

[Ger88] Peter Gerl. Random walks on graphs with a strong isoperimetric property. *J. Theoret. Probab.*, 1(2):171–187, 1988.

[Ger92] S. M. Gersten. Dehn functions and L1-norms of finite presentations. In *Algorithms and Classification in Combinatorial Group Theory*, pages 195–225, Berlin, 1992. Springer.

[Ger93] Steve M. Gersten. Isoperimetric and isodiametric functions of finite presentations. In *Geometric group theory, Vol. 1 (Sussex, 1991)*, volume 181 of *London Math. Soc. Lecture Note Ser.*, pages 79–96. Cambridge Univ. Press, Cambridge, 1993.

[GLS02] Georg Gottlob, Nicola Leone, and Francesco Scarcello. Computing LOGCFL certificates. *Theor. Comput. Sci.*, 270(1-2):761–777, 2002.

[Gri77] R. I. Grigorchuk. Symmetric random walks on discrete groups. *Uspehi Mat. Nauk*, 32(6(198)):217–218, 1977.

[Gro93] M. Gromov. Asymptotic invariants of infinite groups. In *Geometric group theory, Vol. 2 (Sussex, 1991)*, volume 182 of *London Math. Soc. Lecture Note Ser.*, pages 1–295. Cambridge Univ. Press, Cambridge, 1993.

[GS66] Seymour Ginsburg and Edwin H. Spanier. Semigroups, Presburger formulas and languages. *Pacific Journal of Mathematics*, 16:285–296, 1966.

[GS09] Dima Grigoriev and Vladimir Shpilrain. Authentication from matrix conjugation. *Groups Complexity Cryptology*, 1:199–205, 2009.

[HAB02] William Hesse, Eric Allender, and David A. Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *Journal of Computer and System Sciences*, 65:695–716, 2002.

[Hal98] Marshall Hall, Jr. *Combinatorial theory*. Wiley Classics Library. John Wiley & Sons Inc., New York, second edition, 1998.

[Hes01] William Hesse. Division is in uniform TC^0. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *ICALP*, volume 2076 of *Lecture Notes in Computer Science*, pages 104–114. Springer, 2001.

[HF94] K. J. Horadam and G. E. Farr. The conjugacy problem for HNN extensions with infinite cyclic associated groups. *Proc. Amer. Math. Soc.*, 120(4):1009–1015, 1994.

[HL09] Niko Haubold and Markus Lohrey. Compressed word problems in HNN-extensions and amalgamated products. In Anna E. Frid, Andrey Morozov, Andrey Rybalchenko, and Klaus W. Wagner, editors, *CSR*, volume 5675 of *Lecture Notes in Computer Science*, pages 237–249. Springer, 2009.

[HL11] Niko Haubold and Markus Lohrey. Compressed word problems in HNN-extensions and amalgamated products. *Theory Comput. Syst.*, 49(2):283–305, 2011.

[HN01] J.K. Hunter and B. Nachtergaele. *Applied Analysis*. World Scientific Publishing Company Incorporated, 2001.

[HNN49] Graham Higman, Bernhard Neumann, and Hanna Neumann. Embedding theorems for groups. *J. London Math. Soc.*, 24:247–254, 1949.

[Hor81] K. J. Horadam. The word problem and related results for graph product groups. *Proc. American Mathematical Society*, 82:407–408, 1981.

[Hor84] K. J. Horadam. The conjugacy problem for graph products with central cyclic edge groups. *Proc. Amer. Math. Soc.*, 91(3):345–350, 1984.

[Hor89] K. J. Horadam. The conjugacy problem for finite graph products. *Proc. Amer. Math. Soc.*, 106(3):589–592, 1989.

[Jan88] Matthias Jantzen. *Confluent String Rewriting*, volume 14 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1988.

[Kes59a] Harry Kesten. Full Banach mean values on countable groups. *Math. Scand.*, 7:146–156, 1959.

[Kes59b] Harry Kesten. Symmetric random walks on groups. *Trans. Amer. Math. Soc.*, 92:336–354, 1959.

[KL80] Ravindran Kannan and Richard J. Lipton. The orbit problem is decidable. In Raymond E. Miller, Seymour Ginsburg, Walter A. Burkhard, and Richard J. Lipton, editors, *STOC*, pages 252–261. ACM, 1980.

[KL86] R. Kannan and R. J. Lipton. Polynomial-time algorithm for the orbit problem. *J. Assoc. Comput. Mach.*, 33(4):808–821, 1986.

[KM02] I. Kapovich and A. G. Miasnikov. Stallings foldings and subgroups of free groups. *J. Algebra*, 248:608–668, 2002.

[KMP77] D. Knuth, J. H. Morris, and V. Pratt. Fast pattern matching in strings. *SIAM J. Comput.*, 6:323–350, 1977.

[KMSS03] I. Kapovich, A. G. Miasnikov, P. Schupp, and V. Shpilrain. Generic-case complexity, decision problems in group theory and random walks. *J. Algebra*, 264:665–694, 2003.

[KMSS05] I. Kapovich, A. Myasnikov, P. Schupp, and V. Shpilrain. Average-case complexity and decision problems in group theory. *Adv. Math.*, 190:343–359, 2005.

[Lai67] Richard A. Laing. Realization and complexity of commutative events. Technical report, University of Michigan, 1967.

[Lev07] Gilbert Levitt. On the automorphism group of generalized Baumslag-Solitar groups. *Geom. Topol.*, 11:473–515, 2007.

[Lev14] Gilbert Levitt. Quotients and subgroups of Baumslag-Solitar groups. *J. Group Theory*, 2014. Ahead of print.

[Lip66] Seymour Lipschutz. Generalization of Dehn's result on the conjugacy problem. *Proc. Amer. math. Soc.*, 17:759–762, 1966.

[Loc89] Jody Meyer Lockhart. An HNN-extension with cyclic associated subgroups and with unsolvable conjugacy problem. *Trans. Amer. Math. Soc.*, 313(1):331–345, 1989.

[Loc92] Jody Meyer Lockhart. The conjugacy problem for graph products with infinite cyclic edge groups. *Proc. Amer. Math. Soc.*, 114(3):603–606, 1992.

[Loc93] Jody Meyer Lockhart. The conjugacy problem for graph products with finite cyclic edge groups. *Proc. Amer. Math. Soc.*, 117(4):897–898, 1993.

[Loh06] Markus Lohrey. Word problems and membership problems on compressed words. *SIAM J. Comput.*, 35(5):1210–1240, 2006.

[LS01] Roger Lyndon and Paul Schupp. *Combinatorial Group Theory*. Classics in Mathematics. Springer, 2001. First edition 1977.

[LS07] Markus Lohrey and Saul Schleimer. Efficient computation in groups via compression. In Volker Diekert, Mikhail V. Volkov, and Andrei Voronkov, editors, *CSR*, volume 4649 of *Lecture Notes in Computer Science*, pages 249–258. Springer, 2007.

[LZ77] Richard J. Lipton and Yechezkel Zalcstein. Word problems solvable in logspace. *Journal of the Association for Computing Machinery*, 24:522–526, 1977.

[Mag32] Wilhelm Magnus. Das Identitätsproblem für Gruppen mit einer definierenden Relation. *Math. Ann.*, 106:295–307, 1932.

[Mal40] A. Malcev. On isomorphic matrix representations of infinite groups. *Rec. Math. [Mat. Sbornik] N.S.*, 8 (50):405–422, 1940.

[Mal58] Anatolij I. Malcev. On homomorphisms of finite groups. *Ivano Gosudarstvennyi Pedagogicheskii Institut Uchenye Zapiski*, 18:49–60, 1958.

[Mat73] Yuri Matiyasevich. Real-time recognition of the inclusion relation. *Journal of Soviet Mathematics*, 1:64–70, 1973. Translated from Zapiski Nauchnykh Seminarov Leningradskogo Otdeleniya Matematicheskogo Instituta im. V. A. Steklova Akademii Nauk SSSR, Vol. 20, pp. 104–114, 1971.

[Mes72] Stephen Meskin. Nonresidually finite one-relator groups. *Trans. Amer. Math. Soc.*, 164:105–114, 1972.

[Mil71] C. F. Miller III. *On group-theoretic decision problems and their classification*, volume 68 of *Annals of Mathematics Studies*. Princeton University Press, 1971.

[MKS66] Wilhelm Magnus, Abraham Karrass, and Donald Solitar. *Combinatorial Group Theory*. Interscience Publishers (New York), 1966. Reprint of the 2nd edition (1976): 2004.

[MM82] Ernst W. Mayr and Albert R. Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. *Advances in Math.*, 46:305–329, 1982.

[MR08] Alexei G. Myasnikov and Alexander N. Rybalov. Generic complexity of undecidable problems. *J. Symbolic Logic*, 73(2):656–673, 2008.

[MRF12] A.G. Myasnikov, V.N. Remeslennikov, and E.V. Frenkel. Amalgamated products of groups: measures of random normal forms. *Journal of Mathematical Sciences*, 185(2):300–320, 2012.

[MSU08] A. Myasnikov, V. Shpilrain, and A. Ushakov. *Group-based Cryptography*. Advanced courses in mathematics, CRM Barcelona. Birkhäuser Basel, 2008.

[MUW11] Alexei G. Myasnikov, Alexander Ushakov, and Dong Wook Won. The Word Problem in the Baumslag group with a non-elementary Dehn function is polynomial time decidable. *Journal of Algebra*, 345:324–342, 2011.

[MUW12] Alexei G. Myasnikov, Alexander Ushakov, and Dong Wook Won. Power circuits, exponential algebra, and time complexity. *IJAC*, 22, 2012. 51 pages.

[Nor92] S. Northshield. Cogrowth of regular graphs. *Proc. Amer. Math. Soc.*, 116(1):203–205, 1992.

[Nov55] P. S. Novikov. On the algorithmic unsolvability of the word problem in group theory. *Trudy Mat. Inst. Steklov*, pages 1–143, 1955. In Russian.

[OW07] Ronald Ortner and Wolfgang Woess. Non-backtracking random walks and cogrowth of graphs. *Canad. J. Math.*, 59(4):828–844, 2007.

[Pap94] Christos Papadimitriou. *Computation Complexity*. Addison-Wesley, 1994.

[Pla04] A. N. Platonov. An isoparametric function of the Baumslag-Gersten group. *(Russian) Vestnik Moskov. Univ. Ser. I Mat. Mekh.*, 3:12–17, 2004. Russian. Engl. transl. Moscow Univ. Math. Bull. 59 (3) (2004), 12–17.

[Pop02] V. Yu. Popov. On the complexity of the word problem for finitely presented commutative semigroups. *Sibirsk. Mat. Zh.*, 43(6):1339–1349, 2002.

[Rob93] David Robinson. *Parallel Algorithms for Group Word Problems*. PhD thesis, University of California, San Diego, 1993.

[Sch86] A. Schrijver. *Theory of Linear and Integer Programming*. Wiley-Interscience, 1986.

[Sch08] Saul Schleimer. Polynomial-time word problems. *Commentarii Mathematici Helvetici*, 83:741–765, 2008.

[Ser77] Jean-Pierre Serre. *Arbres, amalgames,* SL_2. Société Mathématique de France, Paris, 1977. Avec un sommaire anglais, Rédigé avec la collaboration de Hyman Bass, Astérisque, No. 46.

[Ser80] Jean-Pierre Serre. *Trees*. Springer, 1980.

[Sim79] Hans-Ulrich Simon. Word problems for groups and contextfree recognition. In *Proceedings of Fundamentals of Computation Theory (FCT'79), Berlin/Wendisch-Rietz (GDR)*, pages 417–422. Akademie-Verlag, 1979.

[Sip96] Michael Sipser. *Introduction to the Theory of Computation*. International Thomson Publishing, 1st edition, 1996.

[Soa94] Paolo M. Soardi. *Potential theory on infinite networks*, volume 1590 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1994.

[Sta71] John Stallings. *Group theory and three-dimensional manifolds*. Yale University Press, New Haven, Conn., 1971.

[Sti82] John Stillwell. The word problem and the isomorphism problem for groups. *Bull. Amer. Math. Soc. (N.S.)*, 6(1):33–56, 1982.

[Sud77] Ivan Hal Sudborough. Time and Tape Bounded Auxiliary Pushdown Automata. In Jozef Gruska, editor, *MFCS*, volume 53 of *Lecture Notes in Computer Science*, pages 493–503. Springer, 1977.

[Sud78] Ivan Hal Sudborough. On the Tape Complexity of Deterministic Context-Free Languages. *Journal of the Association for Computing Machinery*, 25:405–414, 1978.

[SZ06] V. Shpilrain and G. Zapata. Combinatorial group theory and public key cryptography. *Appl. Algebra Engrg. Comm. Comput.*, 17:291–302, 2006.

[Taĭ68] M. A. Taĭclin. Algorithmic problems for commutative semigroups. *Dokl. Akad. Nauk SSSR*, 178:786–789, 1968.

[Tre88] Carol Tretkoff. Complexity, combinatorial group theory and the language of palutators. *Theoret. Comput. Sci.*, 56(3):253–275, 1988.

[Ven91] H. Venkateswaran. Properties that characterize LOGCFL. *J. Comput. Syst. Sci.*, 43(2):380–404, 1991.

[Vol99] Heribert Vollmer. *Introduction to Circuit Complexity.* Springer, Berlin, 1999.

[vzGS78] Joachim von zur Gathen and Malte Sieveking. A bound on solutions of linear integer equalities and inequalities. *Proc. Amer. Math. Soc.*, 72(1):155–158, october 1978.

[Waa81] Stephan Waack. Tape complexity of word problems. In Ferenc Gécseg, editor, *Proceedings of Fundamentals of Computation Theory (FCT'81)*, volume 117 of *Lecture Notes in Computer Science*, pages 467–471. Springer, 1981.

[Waa90] Stephan Waack. The parallel complexity of some constructions in combinatorial group theory. *Journal of Information Processing and Cybernetics*, 26(5-6):265–281, 1990.

[Why98] K.M. Whyte. *Discrete Metric Spaces and Coarse Characteristic Classes.* University of Chicago, Department of Mathematics, 1998.

[Wis00] Daniel T. Wise. Subgroup separability of graphs of free groups with cyclic edge groups. *Q. J. Math.*, 51:107–129, 2000.

[Woe00] Wolfgang Woess. *Random Walks on Infinite Graphs and Groups.* Cambridge University Press, 2000.

Index