FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Žaneta Semanišinová

Higher commutators in loop theory

Department of Algebra

Supervisor of the master thesis:	doc. RNDr. David Stanovský, Ph.D.
Study programme:	Mathematics
Study branch:	Mathematical Structures

Prague 2021

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In .... date ....

Author's signature

I would like to thank my supervisor David Stanovský for long discussions about the problems in the thesis, hints on solving them and many useful comments on the text of the thesis. Also, I would like to thank Michael Kompatscher for reading the thesis and for helpful tips and comments. Finally, I would like to thank Jakub Genči for helping me to get back to programming and for the tips on improving the program.

Title: Higher commutators in loop theory

Author: Žaneta Semanišinová

Department: Department of Algebra

Supervisor: doc. RNDr. David Stanovský, Ph.D., Department of Algebra

Abstract: The thesis deals with supernilpotence in loops, building on three equivalent definitions of higher commutators in Mal'tsev algebras due to Aichinger and Mudrinski [2], Bulatov [4] and Opršal [9]. In the thesis, we study identities that occur in 1-, 2- and 3-supernilpotent loops. We prove that a k-supernilpotent loop has a k-nilpotent multiplication group. Moreover, we present results of our implementation of algorithmic testing of supernilpotence in non-associative loops of small orders.

Keywords: loop theory, higher commutator, supernilpotence, nilpotence

Contents

Introduction

Commutator theory is a part of universal algebra, which generalizes group-theoretic definitions such as the commutator of two subgroups, abelianess, or nilpotence to arbitrary algebras. Its development started in 1970s with results of Gumm [7] and Smith [10] and it proved to be very useful not only in varieties closely related to groups, but also in algebras that are very far from them. The work of Freese and McKenzie [5] shows that the commutator theory works particularly well in congruence modular varieties, and hence, in particular, Mal'tsev varieties.

As the name suggests, the binary commutator of two congruences plays central role in the theory. The commutator is defined by a condition for terms of an algebra. In [4], Bulatov generalized the term condition that defines commutator of two congruences to any number of congruences and introduced higher commutators of arbitrary arity. Using these, it is possible to define a new notion of supernilpotence. In Mal'tsev algebras, supernilpotence implies nilpotence and can be defined in different ways, which provide various methods for studying properties of supernilpotent algebras.

Loops are algebras with a binary operation $\cdot$, which is uniquely divisible from both sides, and has a neutral element 1. They naturally generalize groups dropping associativity from the axioms (see Subsection 1.1.1). Therefore, it is not surprising that many group definitions were generalized for loops, such as normal subloops, center or nilpotence. On the other hand, loops are Mal'tsev algebras and hence a lot of universal-algebraic concepts such as commutator theory work well in loops. The arising definitions coincide with the ones based on the group theory and hence we can use universal-algebraic tools to study loop properties.

The binary commutator and related properties for loops were already studied by Stanovský and Vojtěchovský in [12, 11]. Most importantly, they defined a commutator of two normal subloops based on the binary commutator of congruences and described it in loop-theoretical notions, using the inner mapping group of a loop (see Definition 1.2) and a similar notion of total inner mapping group.

In this thesis, we continue to study commutator theory in loops, in particular, we focus on supernilpotent loops. We use three different equivalent definitions of higher commutators for Mal'tsev algebras due to Bulatov [4], Aichinger and Mudrinski [2], and Opršal [9], which allow us to use diverse approaches to study commutators of higher arity and supernilpotence in loops.

In the first chapter, we introduce basic notions and important results from loop theory (Section 1.1) and commutator theory (Section 1.2), which will be used throughout the thesis or which motivated the research. For two results that were particularly significant for us, we also give proofs in modern notation and with more details than in the original papers. The first of them is Bruck's theorem that states that a loop with a k-nilpotent multiplication group is k-nilpotent (Theorem 1.10) and the second is Wright's theorem that shows that a finite loop is supernilpotent if and only if its multiplication group is nilpotent (Theorem 1.13, after reformulation in universal-algebraic notions).

The second chapter is devoted to theoretical results achieved in the thesis. In Section 2.1, we give a simple proof of the fact that supernilpotent loops are nilpotent, since the general proof for Mal'tsev algebras is fairly technical. In Section 2.2, we study the identities that give necessary or sufficient conditions for loops to be supernilpotent based on the inner mappings. In Section 2.3, we prove a stronger version of the forward implication of Wright's theorem [14], i. e., a k-supernilpotent loop has a k-nilpotent multiplication group.

In the third chapter we use Opršal's definition of supernilpotence [9] to develop an algorithm for testing k-supernilpotence in finite loops. In Section 3.1 and 3.2 we describe the algorithm and data representation used in the implementation. We focused on testing 3-supernilpotence, since 1- and 2-supernilpotent loops are shown to be groups in Section 2.2. We ran the 3-supernilpotence test on all non-associative 8-element loops with multiplication group of nilpotence class 3 and all non-associative 9-element loops. The results are summarized in Section 3.3. In Section 3.4, we draw conclusions from them and discuss the implications for further research.

The thesis contains original results (Proposition 2.1, Theorem 2.4) that are presented in Chapter 2 including necessary conditions for 3-supernilpotent loops based on inner mappings and a necessary condition for k-supernilpotent loops based on the nilpotence class of their multiplication group. Furthermore, in Chapter 3 we present an algorithm for testing supernilpotence and the results of the tests performed by the implemented algorithm.

1. Preliminaries

The purpose of the thesis is to study properties that arise from the universal-algebraic commutator theory in context of loops. In the first chapter we introduce the loop-theoretical and universal-algebraic notions that are going to be used throughout the text and present the well-known results which we build on or which motivated the research.

1.1 Loops

In this section we deal with basic loop-theoretical notions, including nilpotence, which is generalized from the well-known notion in groups. We list some basic properties and present detailed proofs of two theorems (Theorem 1.10 and 1.13) that motivated us to study the connection between supernilpotence of a loop and nilpotence of its multiplication group.

1.1.1 Basic notions

Definition 1.1 (loop)**.** *Let Q be a set equipped with a binary operation $\cdot : Q \times Q \to Q$. We call $(Q, \cdot)$ a* quasigroup *if for every $a, b \in Q$ there exist unique $x, y \in Q$ such that $a \cdot x = b$ and $y \cdot a = b$. We denote these by $a\backslash b$ and b/a respectively.*

If there is an element $1 \in Q$ such that for every $a \in Q$ it holds that $a \cdot 1 = 1 \cdot a = a$, then we say that $(Q, \cdot, 1)$ is a loop*.*

Note that every group is a loop and a loop is a group if and only if the binary operation $\cdot$ is associative.

Observe that $\backslash$ and $/$ can be understood as binary operations on Q. Using this, a loop can be alternatively defined as a universal algebra $(Q, \cdot, \backslash, /, 1)$ satisfying the identitities

$$x\backslash(x \cdot y) = y, \quad x \cdot (x\backslash y) = y, \quad (y \cdot x)/x = y, \quad (y/x) \cdot x = y, \quad x \cdot 1 = x = 1 \cdot x.$$

It is easy to see that the two definitions are equivalent.

Definition 1.2. *Let Q be a loop. For every $x \in Q$, let L_x, R_x, be permutations of Q defined by*

$$L_x(y) = xy, \qquad R_x(y) = yx.$$

The mappings L_x and R_x are called left *and* right translations*.*

The group generated by left and right translations of Q is called the multiplication group *of Q and denoted $\mathrm{Mlt}(Q)$. The stabilizer of 1 in $\mathrm{Mlt}(Q)$ is called the* inner mapping group *of Q and denoted $\mathrm{Inn}(Q)$.*

Observe that we have $L_x^{-1}(y) = x\backslash y$ and $R_x^{-1}(y) = y/x$.

Let us consider the following mappings

$$L_{x,y} = L_{xy}^{-1} L_x L_y, \quad R_{x,y} = R_{yx}^{-1} R_x R_y, \quad T_x = R_x^{-1} L_x.$$

The following well-known proposition shows why the mentioned mappings are interesting. The proof can be found for example in [12].

Proposition 1.3. *Let Q be a loop. Then*

$$\mathrm{Inn}(Q) = \langle L_{x,y},\ R_{x,y},\ T_x :\ x, y \in Q\rangle.$$

Next we define a normal subloop, which is analogous to the concept of normal subgroup of a group.

Definition 1.4 (normal subloop)**.** *A subloop N of a loop Q is said to be* normal*, if*

$$xN = Nx, \quad x(yN) = (xy)N, \quad N(xy) = (Nx)y,$$

for all $x, y \in Q$.

Observe that the sets xN, $x \in Q$ have the same size and form a partition of Q, hence for a finite Q, $|N|$ divides $|Q|$.

Normal subloops of Q correspond to congruences of Q in a natural way. If N is a normal subloop of Q, then we can define a binary relation α_N by

$$a\ \alpha_N\ b \text{ iff } a/b \in N$$

or, equivalently, $a\backslash b \in N$, $b/a \in N$ or $b\backslash a \in N$ (the equivalence is clear after observing that any of these conditions is equivalent to $aN = Na = Nb = bN$). Using the mentioned observation, it is straightforward to verify that α_N is a congruence. Similarly, if α is a congruence on Q, we define a subset

$$N_\alpha = \{a \in Q \mid a\ \alpha\ 1\},$$

which can be easily shown to be a normal subloop of Q using that, for $a \in Q$, the α-block of a is precisely $aN_\alpha = N_\alpha a$. Maps $N \mapsto \alpha_N$ and $\alpha \mapsto N_\alpha$ give a bijective correspondence between normal subloops and congruences of Q.

Another well-known characterization of normal subloops uses inner mapping group.

Proposition 1.5. *Let N be a subloop of a loop Q. The following conditions are equivalent:*

1. *N is normal,*
2. *$f(N) \subseteq N$ for every $f \in \mathrm{Inn}(Q)$,*
3. *$f(N) = N$ for every $f \in \mathrm{Inn}(Q)$,*
4. *N is the block containing 1 of some congruence of Q.*

1.1.2 Nilpotence

Similarly as in the group theory, nilpotence in loops is defined using the center of a loop.

Definition 1.6 (center of a loop)**.** *Let Q be a loop. The* center *of Q is the set $Z(Q)$ of all elements $a \in Q$ such that for all $x, y \in Q$ the following holds:*

- *$a(xy) = (ax)y$,*

- $x(ay) = (xa)y$,
- $x(ya) = (xy)a$,
- $ax = xa$.

$Z(Q)$ is clearly a normal subloop of Q.

Definition 1.7 (nilpotent loop)**.** *Let Q be a loop. We define the* iterated centers *of Q as normal subloops $Z_i(Q)$, $i = 0, 1, \dots$ such that $Z_0(Q) = 1$ and $Z_{i+1}(Q)/Z_i(Q) = Z(Q/Z_i(Q))$. Q is said to be k-*nilpotent *if $Z_k(Q) = Q$ for some $k \geq 0$. The class of nilpotence is the smallest k satisfying that Q is k-nilpotent.*

Observe that $Z_i(Q)$ is a normal subloop for every $i \geq 0$ (this is proved by induction since $Z_{i+1}(Q)$ is the preimage of a normal subloop under projection on $Q/Z_i(Q)$) and hence the definition is correct. Moreover, observe that if a loop is k-nilpotent for some k, then it is l-nilpotent for any $l \geq k$.

We say that a loop is *nilpotent of class k*, if it is k-nilpotent, but not $(k-1)$-nilpotent. If there exists a k such that a loop is k-nilpotent, we say that it is *nilpotent*.

We prove a very useful equivalent definition of loop nilpotence in the following proposition. First we need an auxiliary lemma.

Lemma 1.8. *Let Q be a loop and $N_1 \subseteq N_2$ be two of its normal subloops. For every $a \in Q$, if $aN_1 \in Z(Q/N_1)$, then $aN_2 \in Z(Q/N_2)$.*

Proof. It suffices to prove the claim for normal subloops $1 \subseteq N$, then by the choice $Q = Q/N_1$, $N = N_2/N_1$ and using that

$$(aN_1)N_2/N_1 \in Z((Q/N_1)/(N_2/N_1)) \quad \text{iff} \quad aN_2 \in Z(Q/N_2),$$

the general claim follows.

Consider $a \in Z(Q)$ and a normal subloop N in Q. We want to prove that $aN \in Z(Q/N)$. Consider $xN, yN \in Z(Q/N)$. Then

$$(aN)((xN)(yN)) = a(xy)N = (ax)yN = ((aN)(xN))(yN).$$

The other conditions are proved similarly. □

Proposition 1.9. *A loop is 1-nilpotent if and only if it is an abelian group and if $k > 1$, a loop Q is k-nilpotent if and only if the loop $Q/Z(Q)$ is $(k-1)$-nilpotent.*

Proof. A loop Q is 1-nilpotent if and only if $Q = Z(Q)$, which is if and only if it is associative and commutative, that is, an abelian group.

To deal with the general case, we start with a reformulation of the definition of nilpotence with slightly relaxed condition on the series.

Claim. A loop Q is k-nilpotent if and only if there exist normal subloops of Q

$$1 = Q_0 \leq Q_1 \leq \cdots \leq Q_k = Q$$

such that $Q_{i+1}/Q_i \leq Z(Q/Q_i)$, $i = 0, \dots k-1$.

Proof. Nilpotent loops clearly satisfy the condition with a choice $Q_i = Z_i(Q)$. For the other implication, it suffices to show that $Q_i \leq Z_i(Q)$, $i = 0, 1, \dots, k$. This is true for $i = 0$. Suppose that $Q_i \leq Z_i(Q)$ for some $i \geq 0$, then for $a \in Q_{i+1}$, we have by Lemma 1.8 that

$$aQ_i \in Q_{i+1}/Q_i \subseteq Z(Q/Q_i) \Rightarrow aZ_i(Q) \in Z(Q/Z_i(Q)) = Z_{i+1}(Q)/Z_i(Q)$$

and hence $a \in Z_{i+1}(Q)$, so the claim follows by induction.

Now we are ready to prove the proposition. Suppose first that we have a k-nilpotent loop Q, $k > 1$. Let us have series of normal subloops

$$1 = Q_0 \leq Q_1 \leq \cdots \leq Q_k = Q$$

such that $Q_{i+1}/Q_i \leq Z(Q/Q_i)$, $i = 0, \dots k-1$. Then $Q_1 \leq Z(Q)$, let $j > 1$ be the smallest such that $Q_j > Z(Q)$. We claim that

$$Z(Q)/Z(Q) \leq Q_j/Z(Q) \leq Q_{j+1}/Z(Q) \leq \cdots \leq Q_k/Z(Q) = Q/Z(Q)$$

is a series certifying that nilpotence class of $Q/Z(Q)$ is at most $k-1$. We need to verify that the quotient of two consecutive members of the series is included in the corresponding center. Using second isomorphism theorem, this condition is equivalent to

$$Q_j/Z(Q) \leq Z(Q/Z(Q)) \quad \text{and} \quad Q_{i+1}/Q_i \leq Z(Q/Q_i) \text{ for } i = j, \dots, k-1.$$

The former follows from Lemma 1.8 for normal subloops $Q_{j-1} \leq Z(Q)$ using $Q_j/Q_{j-1} \leq Z(Q/Q_{j-1})$ and the latter follows from the definition of the series $Q_0 \leq Q_1 \leq \cdots \leq Q_k$.

For the reverse implication, suppose that $Q/Z(Q)$ is $(k-1)$-nilpotent. Then there is a series

$$Z(Q)/Z(Q) = Q_0/Z(Q) \leq Q_1/Z(Q) \leq \cdots \leq Q_{k-1}/Z(Q) = Q/Z(Q),$$

such that by second isomorphism theorem $Q_{i+1}/Q_i \leq Z(Q/Q_i)$, $i = 0, \dots, k-2$. The series

$$1 \leq Z(Q) = Q_0 \leq Q_1 \leq \cdots \leq Q_{k-1} = Q$$

then certifies that Q is k-nilpotent. $\square$

In groups, the loop definition of nilpotence coincides with the traditional definition. Hence groups of prime power order are nilpotent and finite nilpotent groups decompose as a direct product of groups of prime power order. Neither statement is true for loops in general. For example,

- every non-associative loop of prime order is not nilpotent, since $|Z(Q)|$ divides $|Q|$,
- there is a directly indecomposable nilpotent loop of order 6 [8, Sec. 5].

For a nilpotent loop Q, various interesting properties of $\mathrm{Mlt}(Q)$ have been proved by Bruck [3]. It turns out that there is a strong connection between the nilpotence of a loop and properties of its multiplication group.

Theorem 1.10. [3, Chap. 8] *Let Q be a loop. If $\mathrm{Mlt}(Q)$ is k-nilpotent, then Q is k-nilpotent.*

The theorem was originally presented by Bruck just for finite loops. However, his proof does not use finiteness of the loop and hence it holds in general. We present here Bruck's proof translated to modern notation. The proof is based on the following proposition.

Proposition 1.11. [3, Theorem 8B] *Let Q be a loop and $Z_i = Z_i(Q)$, $i = 0, 1, \dots$, be its central series. Define an ascending series $\mathcal{N}_i$ of subgroups of $\mathrm{Mlt}(Q)$ as follows: $\mathcal{N}_0 = \mathrm{Inn}(Q)$ and $\mathcal{N}_{i+1}$ is the normalizer of $\mathcal{N}_i$ in $\mathrm{Mlt}(Q)$, $i = 0, 1, \dots$ Then, for every i,*

$$\mathcal{N}_i = \{R_a f \mid f \in \mathrm{Inn}(Q), a \in Z_i\}.$$

In particular, Q is nilpotent of class k if and only if $\mathcal{N}_k = \mathrm{Mlt}(Q)$ and $\mathcal{N}_{k-1} \neq \mathrm{Mlt}(Q)$.

Proof. We prove the claim by induction on i. For $i = 0$, this is certainly true, so suppose that the claim holds for some $i \geq 0$ and we prove it for $i + 1$.

Let $t \in \mathrm{Mlt}(Q)$, then $t = R_x h$, where $x = t(1)$ and $h \in \mathrm{Inn}(Q)$. Then, by the induction hypothesis, $t \in \mathcal{N}_{i+1}$ if and only if

$$\forall a \in Z_i, f \in \mathrm{Inn}(Q)\ \exists b \in Z_i, g \in \mathrm{Inn}(Q) \text{ such that } (R_a f)(R_x h) = (R_x h)(R_b g) \qquad (1.1)$$

Suppose now that $t \in \mathcal{N}_{i+1}$, we need to show that $x \in Z_{i+1}$. Choose a and f and find corresponding b and g. By applying (1.1) on $1 \in Q$, we obtain

$$f(x) \cdot a = h(b) \cdot x.$$

Since $b \in Z_i$ and $h \in \mathrm{Inn}(Q)$, $h(b) \in Z_i$. Since $a \in Z_i$ was arbitrary, it follows from the normality of Z_i that $f(x)Z_i = xZ_i$. By setting f to suitable inner mappings (namely $L_{y,z}, R_{y,z}, [L_y, R_z]$ and T_y), we obtain that $xZ_i \in Z(Q/Z_i)$, that is, $x \in Z_{i+1}$.

To prove the reverse inclusion, let $t = R_x h$, $f \in \mathrm{Inn}(Q)$, $x \in Z_{i+1}$ and consider $a \in Z_i$, $f \in \mathrm{Inn}(Q)$. Then $R_a f t = R_y h_1$, where $h_1 \in \mathrm{Inn}(Q)$ and $y = R_a f t(1) = f(x) \cdot a$. Since $x \in Z_{i+1}$, $f(x)Z_i = xZ_i = Z_i x$ (it holds for $f = L_{y,z}, R_{y,z}, T_y$ from the definition of a center and, by Proposition 1.3, this implies that it holds for all $f \in \mathrm{Inn}(Q)$). Therefore, $y = f(x) \cdot a = cx$ for some $c \in Z_i$.

Now, for $h_2 = R_c^{-1} R_x^{-1} R_{cx} h_1 \in \mathrm{Inn}(Q)$,

$$R_a f t = R_{cx} h_1 = R_x R_c h_2 = (R_x h)(h^{-1} R_c h_2) = (R_x h)(R_b g),$$

for $b = h^{-1}(c) \in Z_i$ and some $g \in \mathrm{Inn}(Q)$. It follows that $t \in \mathcal{N}_{i+1}$.

Now it is straightforward to verify that, for every i, $Z_i = Q$ if and only if $\mathcal{N}_i = \mathrm{Mlt}(Q)$, which completes the proof.

□

Proof of Theorem 1.10. Let us denote $Z_i(Q) = Z_i$ and $Z_i(\mathrm{Mlt}(Q)) = \mathcal{Z}_i$, $i = 0, 1, \dots$ By Proposition 1.11, it suffices to show that, for every i, $\mathcal{N}_i \geq \mathcal{Z}_i$. We proceed by induction on i, for $i = 0$, this is trivial. Let $\mathcal{N}_i \geq \mathcal{Z}_i$ for some i, we show that then $\mathcal{N}_{i+1} \geq \mathcal{Z}_{i+1}$.

Let $f \in \mathcal{Z}_{i+1}$, then $f\mathcal{Z}_i \in Z(\mathrm{Mlt}(Q)/\mathcal{Z}_i)$ and for every $g \in \mathcal{N}_i$,

$$fgf^{-1}\mathcal{Z}_i = g\mathcal{Z}_i \Rightarrow fgf^{-1}\mathcal{N}_i = g\mathcal{N}_i = \mathcal{N}_i \Rightarrow fgf^{-1} \in \mathcal{N}_i.$$

It follows that f is in the normalizer of $\mathcal{N}_i$, i.e., $\mathcal{N}_{i+1}$. The claim now follows. □

For finite loops, there is even more than can be proved.

Theorem 1.12. [3, Chap. 8][8, Sec. 2] *Let Q be a finite loop.*

1. *If Q is nilpotent, then* $\mathrm{Mlt}(Q)$ *is solvable.*
2. *If Q is nilpotent, then all primes dividing* $|\mathrm{Mlt}(Q)|$ *also divide* $|Q|$.
3. *Let p be a prime number. Then Q is nilpotent of p-power size if and only if* $\mathrm{Mlt}(Q)$ *is a p-group.*

The claim (3) was originally formulated in a weaker version: *A necessary and sufficient condition for a finite loop Q of p-power size to be nilpotent is that the group* $\mathrm{Mlt}(Q)$ *is a p-group.* However, since $\mathrm{Mlt}(Q)$ acts transitively on Q, Q itself is an orbit of this action and hence, $|Q|$ divides $|\mathrm{Mlt}(Q)|$.

The following theorem by Wright will be interpreted in universal-algebraic context in Section 2.3, where we prove a stronger version of the forward implication.

Theorem 1.13. [14, Theorem 1] *A finite loop Q is a direct product of nilpotent loops of prime power order if and only if* $\mathrm{Mlt}(Q)$ *is nilpotent.*

It follows from the theorem that nilpotent loops do not need to have nilpotent multiplication group, consider, for example, the mentioned directly indecomposable nilpotent loop of order 6.

We present here a slightly simplified proof of the theorem in modern notation and including more details. We start with two simple lemmas.

Lemma 1.14. *Let I be a set and Q be a direct product of loops Q_i, $i \in I$. Then* $\mathrm{Mlt}(Q)$ *is a direct product of groups* $\mathrm{Mlt}(Q_i)$.

Proof. For every $x = (x_i : i \in I) \in Q$, the left translation L_x is of the form $(L_{x_i} : i \in I)$ and similarly for the right translation R_x. Since $\mathrm{Mlt}(Q)$ is generated by left and right translations, the claim now follows. □

For a subgroup $\mathcal{G}$ of $\mathrm{Mlt}(Q)$, we denote

$$\mathcal{G}(1) = \{g(1) \mid g \in \mathcal{G}\}.$$

If $\mathcal{G}$ is normal in $\mathrm{Mlt}(Q)$, then for every $f \in \mathrm{Inn}(Q)$ and every $g \in \mathcal{G}$

$$f(g(1)) = fgf^{-1}(f(1)) = fgf^{-1}(1) \in \mathcal{G}(1),$$

hence $\mathcal{G}(1)$ is a normal subloop by Proposition 1.5 (2).

For a normal subloop N of Q, we denote

$$N^* = \{f \in \mathrm{Mlt}(Q) \mid \forall x \in Q : f(x) \in Nx\}.$$

Lemma 1.15. [14, Proposition 5] *Let Q be a loop and let* $\mathrm{Mlt}(Q) \simeq \mathcal{P} \times \mathcal{R}$ *where* $\mathcal{P}(1) \cap \mathcal{R}(1) = 1$. *Then* $(\mathcal{P}(1))^* = \mathcal{P}$ *and* $(\mathcal{R}(1))^* = \mathcal{R}$.

Proof. First observe that $\mathcal{P}$ and $\mathcal{R}$ are normal in $\mathrm{Mlt}(Q)$ and hence $\mathcal{P}(1)$ and $\mathcal{R}(1)$ are normal subloops of Q. Moreover, $\mathcal{P} \subseteq (\mathcal{P}(1))^*$, since for every $f \in \mathcal{P}$

$$f(x)/x = R_x^{-1} f R_x(1) \in \mathcal{P}(1).$$

Similarly, $\mathcal{R} \subseteq (\mathcal{R}(1))^*$.

Let $f \in (\mathcal{P}(1))^* \cap (\mathcal{R}(1))^*$, then for every $x \in Q$

$$f(x) \in \mathcal{P}(1)x \cap \mathcal{R}(1)x \Rightarrow f(x)/x \in \mathcal{P}(1) \cap \mathcal{R}(1) = 1.$$

It follows that $(\mathcal{P}(1))^* \cap (\mathcal{R}(1))^* = 1$.

Since $\mathrm{Mlt}(Q) = \mathcal{P} \times \mathcal{R}$, if $\mathcal{P} \subsetneq (\mathcal{P}(1))^*$ or $\mathcal{R} \subsetneq (\mathcal{R}(1))^*$, we would find a non-identity element of $(\mathcal{P}(1))^* \cap (\mathcal{R}(1))^* = 1$. Hence $\mathcal{P} = (\mathcal{P}(1))^*$ and $\mathcal{R} = (\mathcal{R}(1))^*$. □

Proof of Theorem 1.13. ($\Rightarrow$) If $Q = \prod_{i \in I} Q_i$, where Q_i is nilpotent of prime power order for every i, then $\mathrm{Mlt}(Q_i)$ is a p-group by Theorem 1.12 (3), and $\mathrm{Mlt}(Q)$ is a direct product of these p-groups by Lemma 1.14. Hence $\mathrm{Mlt}(Q)$ is nilpotent.

($\Leftarrow$) Proceed by induction on $|Q|$. For $|Q| = 1$ this is trivial. Let $\mathrm{Mlt}(Q) \simeq \mathcal{P} \times \mathcal{R}$ with $\mathcal{P}$ a non-trivial Sylow p-subgroup of $\mathrm{Mlt}(Q)$. Since $\mathrm{Mlt}(Q)$ is nilpotent, $\mathcal{P}$ and $\mathcal{R}$ are also nilpotent.

If $\mathcal{R} = 1$, then $\mathrm{Mlt}(Q)$ is a p-group and the claim follows by Theorem 1.12 (3). Hence for the rest of the proof, we assume $\mathcal{R} \neq 1$.

Let $P = \mathcal{P}(1)$ and $R = \mathcal{R}(1)$ be the orbits of 1. Both P and R are normal subloops of Q. The size of $|P|$ is a power of p since it divides $|\mathcal{P}|$ and $p \nmid |R|$ since it divides $|\mathcal{R}|$. In particular, $P \cap R = 1$.

We are going to prove that $Q = PR$. Since P and R are normal in Q, PR is a subloop of Q. Hence it is enough to show $|Q| = |PR| = |P| \cdot |R|$, where the second equality follows from $P \cap R = 1$.

Lemma 1.15 implies that $P^* = \mathcal{P}$ and $R^* = \mathcal{R}$. In particular, $P, R \neq 1$. It is straightforward to show

$$\mathrm{Mlt}(Q/P) \simeq \mathrm{Mlt}(Q)/P^* \simeq \mathcal{R},$$

which is a nilpotent group, and thus Q/P decomposes as a direct product of nilpotent loops of prime power sizes by the induction assumption. Moreover, none of the primes in the decomposition is equal to p, since this would imply by Theorem 1.12 (3) and Lemma 1.14 that p divides $\mathcal{R}$. Likewise,

$$\mathrm{Mlt}(Q/R) \simeq \mathrm{Mlt}(Q)/R^* \simeq \mathcal{P},$$

which is a p-group, and thus Q/R is nilpotent and of p-power size.

Finally, let $|Q| = p^k r$, $p \nmid r$. Since $p \nmid |Q/P|$ and P is of p-power size, we must have $|P| = p^k$. Since Q/R is of p-power size, $|R| \geq r$. Altogether, we must have $|PR| = |P| \cdot |R| = p^k r$.

Now, $Q = PR \simeq P \times R$ by an isomorphism

$$\begin{aligned} f : P \times R &\to PR \\ (p, r) &\mapsto pr \end{aligned}$$

Since $P, R \neq 1$, the claim follows from the induction hypothesis.

□

1.2 Commutator theory

In this section we introduce the notion of a higher commutator of congruences and use it to define supernilpotent algebras. Moreover, we give a definition of a nilpotent algebra, whose definition uses just binary commutators, and present the most important results that show the connection between these two notions.

1.2.1 Higher commutator

In this subsection, we present three different definitions of a higher commutator of congruences. Higher commutators rise as a generalization of widely studied binary commutators and are fundamental for the definition of supernilpotence.

The following property has strong consequences on the properties of algebras not only in commutator theory.

Definition 1.16 (Mal'tsev algebra)**.** *An operation $m : A \to A$ on a set A is said to be* Mal'tsev*, if it satisfies the identity*

$$m(y, x, x) = m(x, x, y) = y.$$

An algebra A is said to be Mal'tsev*, if it possess a* Mal'tsev term*, that is, a ternary term that induces a Mal'tsev operation on A.*

Every loop is Mal'tsev with a term $x(y\backslash z)$. Many of the universal-algebraic results that we build upon were generalized beyond Mal'tsev algebra, but this is not important for us.

There are several definitions of the higher commutator, which are equivalent for Mal'tsev algebras: the original term condition of Bulatov [4], its reformulation by Opršal [9] using forks in certain relations and a bound on essential arity of absorbing polynomials by Aichinger and Mudrinski [2]. Here we introduce these three approaches, since all of them will be used throughout the thesis to study supernilpotence in loops.

Definition 1.17 (Bulatov's term condition)**.** [4] *Let A be an algebra, $\alpha_1, \dots, \alpha_n$, β, δ its congruences. We say that $\alpha_1, \dots, \alpha_n$* centralize β modulo δ *if, for every term operation t and all pairs of tuples $\mathbf{a}_i \, \alpha_i \, \mathbf{b}_i$, $\mathbf{u} \, \beta \, \mathbf{v}$,*

$$\begin{aligned}
&\forall (\mathbf{x}_1, ..., \mathbf{x}_n) \in \{\mathbf{a}_1, \mathbf{b}_1\} \times ... \times \{\mathbf{a}_n, \mathbf{b}_n\} \setminus \{(\mathbf{b}_1, ..., \mathbf{b}_n)\} \\
&t(\mathbf{x}_1, ..., \mathbf{x}_n, \mathbf{u}) \, \delta \, t(\mathbf{x}_1, ..., \mathbf{x}_n, \mathbf{v}) \\
&\qquad\qquad \Downarrow \\
&t(\mathbf{b}_1, ..., \mathbf{b}_n, \mathbf{u}) \, \delta \, t(\mathbf{b}_1, ..., \mathbf{b}_n, \mathbf{v}).
\end{aligned}$$

It turns out that for Mal'tsev algebras, it is enough to consider Bulatov's term condition for single elements instead of tuples.

Proposition 1.18. [2, Proposition 5.4] *Let A be a Mal'tsev algebra, $\alpha_1, \dots, \alpha_n$, β, δ its congruences. Then $\alpha_1, \dots, \alpha_n$ centralize β modulo δ if and only if for every term* *operation t and all elements $a_i\,\alpha_i\,b_i$, $u\,\beta\,v$,*

$$\begin{aligned}
&\forall(x_1,...,x_n)\in\{a_1,b_1\}\times...\times\{a_n,b_n\}\smallsetminus\{(b_1,...,b_n)\}\\
&t(x_1,...,x_n,u)\,\delta\,t(x_1,...,x_n,v)\\
&\qquad\qquad\Downarrow\\
&t(b_1,...,b_n,u)\,\delta\,t(b_1,...,b_n,v).
\end{aligned}$$

The following observation is needed in order to define the commutator.

Observation 1.19. *Let A be an algebra, $\alpha_1,\dots,\alpha_n$, β, δ_i, $i\in I$, its congruences. If $\alpha_1,\dots,\alpha_n$ centralize β modulo δ_i, $i\in I$, then $\alpha_1,\dots,\alpha_n$ centralize β modulo $\bigwedge_{i\in I}\delta_i$.*

Definition 1.20 (commutator of congruences)**.** *Let A be an algebra, $\alpha_1,\dots,\alpha_n$, β its congruences. The $(n+1)$-ary commutator $[\alpha_1,\dots,\alpha_n,\beta]$ is defined as the smallest congruence δ of A such that $\alpha_1,\dots,\alpha_n$ centralize β modulo δ.*

Next we introduce a syntactic reformulation of the term condition due to Opršal [9], which is good for algorithmic calculation of commutators and which will be used in Chapter 3.

Definition 1.21 (fork)**.** *Let A be a set. A* fork *in a relation $R\subseteq A^n$ in coordinate i is a pair (a,b) such that there are $\mathbf{u},\mathbf{v}\in R$ satisfying that $u_j=v_j$ for all $j\neq i$, $u_i=a$, $v_i=b$.*

In the next definition will use the following notation. For a positive integer k, let $k(i)$ denote $(i+1)$-th digit from the right of the binary expansion of k. For a pair (a,b) we will denote $a=(a,b)_{(0)}$ and $b=(a,b)_{(1)}$.

Definition 1.22. [9, Def. 3.1] *Let A be an algebra, and $\alpha_0,\dots,\alpha_{n-1}$ congruences of A. For $a,b\in A$, we define*

$$c_i^n(a,b)=((a,b)_{(k(i))}\mid k<2^n).$$

Then let

$$\Delta(\alpha_0,\dots\alpha_{n-1})=Sg\{c_i^n(a,b)\mid i<n, a\;\alpha_i\;b\}.$$

Example 1.23. The relation $\Delta(\alpha_0,\alpha_1,\alpha_2)$ is generated by the tuples $c_i^3(a,b)$, $i=0,1,2$, that have one of the following forms: (a,b,a,b,a,b,a,b), where $a\;\alpha_0\;b$; (a,a,b,b,a,a,b,b), where $a\;\alpha_1\;b$; or (a,a,a,a,b,b,b,b), where $a\;\alpha_2\;b$.

Proposition 1.24. [9, Theorem 1.2] *Let A be a Mal'tsev algebra and $\alpha_1,\dots,\alpha_n,\beta$ its congruences. Then $[\alpha_1,...,\alpha_n,\beta]$ is the set of all forks in $\Delta(\alpha_1,...,\alpha_n,\beta)$ in the last coordinate.*

An alternative approach to higher commutators uses the notion of absorbing polynomials.

Definition 1.25. *Let A be an algebra, $\mathbf{a},e\in A$. A polynomial operaton f of A is called* absorbing at $\mathbf{a}$ into e *if $f(\mathbf{u})=e$ whenever there is i such that $u_i=a_i$.*

Proposition 1.26. [2, Lemma 6.9] *Let A be a Mal'tsev algebra and $\alpha_1, \dots, \alpha_{n+1}$ its congruences. Then $[\alpha_1, \dots, \alpha_{n+1}]$ is equal to*

$$Cg\{(f(\mathbf{b}), e) : \ f \text{ is a } (n+1)\text{-ary polynomial absorbing at } \mathbf{a} \text{ with value } e, \ \forall i : \ b_i \, \alpha_i \, a_i\}.$$

We list here elementary properties of higher commutators in Mal'tsev algebras. They can be found with proofs in [2] as conditions (HC1)-(HC8).

Proposition 1.27. *Let A be an algebra and $\alpha_1, \alpha_2, \dots \alpha_k$, $k \geq 2$ be congruences of A. Then the following hold:*

1. *$[\alpha_1, \dots, \alpha_k] \leq \bigwedge_{i=1}^{k} \alpha_i$;*
2. *if $\beta_1, \dots \beta_k$ are congruences of A such that $\alpha_i \leq \beta_i$ for $i = 1, \dots, k$ then*
$$[\alpha_1, \dots \alpha_k] \leq [\beta_1, \dots \beta_k];$$
3. *$[\alpha_1, \dots, \alpha_k] \leq [\alpha_2, \dots, \alpha_k]$.*

Furthermore, if A is a Mal'tsev algebra, then we have:

4. *for every permutation π of $\{1, \dots, k\}$, $[\alpha_1, \dots, \alpha_k] = [\alpha_{\pi(1)}, \dots, \alpha_{\pi(k)}]$;*
5. *if δ is a congruence of A, then $[\alpha_1, \dots \alpha_k] \leq \delta$ if and only if $\alpha_1, \dots \alpha_{k-1}$ centralize α_k modulo δ;*
6. *if β is a congruence of A, $\beta \leq \alpha_1, \dots, \alpha_k$, then in A/β we have*
$$[\alpha_1/\beta, \dots, \alpha_k/\beta] = ([\alpha_1, \dots, \alpha_k] \vee \beta)/\beta;$$
7. *if I is a nonempty set, $j \in \{1, \dots, k\}$ and β_i, $i \in I$ are congruences of A, then*
$$[\alpha_1, \dots, \alpha_{j-1}, \bigvee_{i \in I} \beta_i, \alpha_{j+1}, \dots, \alpha_k] = \bigvee_{i \in I} [\alpha_1, \dots, \alpha_{j-1}, \beta_i, \alpha_{j+1}, \dots, \alpha_k];$$
8. *for all $j \in \{2, \dots, k\}$*
$$[\alpha_1, \dots, \alpha_{j-1}, [\alpha_j, \dots, \alpha_k]] \leq [\alpha_1, \dots, \alpha_k].$$

The following proposition follows easily by induction from (8) in the preceding proposition.

Proposition 1.28. *Let A be a Mal'tsev algebra, $\alpha_1, ..., \alpha_{n+1}$ its congruences. Then*

$$[\alpha_1, [\alpha_2, [..., [\alpha_n, \alpha_{n+1}] \dots]]] \leq [\alpha_1, ..., \alpha_{n+1}].$$

1.2.2 Nilpotence and supernilpotence

In rest of the text we focus on the notion of nilpotence and supernilpotence. In this subsection, we introduce universal-algebraic definitions of the notions and present theorems and propositions that illustrate the connection between them.

We will use the following notation: For an algebra A, denote its largest conguence $A \times A$ by 1_A and its smallest congruence $\{(a, a) \mid a \in A\}$ by 0_A. Observe that for a loop Q, the corresponding normal subloops are respectively Q and 1.

Definition 1.29 (nilpotent algebra). *An algebra A is said to be k-nilpotent if*

$$\underbrace{[1_A, [1_A, [..., [1_A, 1_A] \ldots]]]}_{k+1} = 0_A.$$

A loop is k-nilpotent in the sense of universal algebra if and only if it is k-nilpotent in the sense of loop theory as in Definition 1.7 [12, 11].

An algebra satisfying the equivalent conditions of Theorem 1.30 is called *k-supernilpotent.*

Theorem 1.30. *The following are equivalent for a Mal'tsev algebra A:*

1. $[\underbrace{1_A, ..., 1_A}_{k+1}] = 0_A$ *(in the sense of Bulatov's term condition),*
2. $\Delta(\underbrace{1_A, ..., 1_A}_{k+1})$ *contains no fork in the last coordinate,*
3. *every absorbing polynomial of arity $k + 1$ is constant.*

Condition (2) comes from Opršal's fork approach, see Proposition 1.24. Condition (3) comes from Aichinger and Mudrinski, see Proposition 1.26. We say that an algebra is supernilpotent of class k, if it is k-supernilpotent, but not $(k - 1)$-supernilpotent. If there exists a k such that an algebra is k-supernilpotent, we say that it is *supernilpotent.*

1-supernilpotent algebras are called *abelian* and have strong properties, especially under Mal'tsev assumption.

The following observation is useful for the absorbing polynomial approach to supernilpotence.

Observation 1.31. *Assume that every k-absorbing polynomial of algebra A is constant. Then, for all $l > k$, every l-absorbing polynomial of A is constant.*

In the finite case, there is another useful characterization, which will be used to reformulate Theorem 1.13.

Theorem 1.32. [2, Sec. 7] *The following are equivalent for a finite Mal'tsev algebra A of finite signature:*

1. *A is supernilpotent,*
2. *A is a direct product of nilpotent algebras of prime power size.*

For Mal'tsev algebras, supernilpotence is a stronger notion than nilpotence.

Proposition 1.33. [2] *If a Mal'tsev algebra is k-supernilpotent then it is k-nilpotent.*

Proof. Follows from Proposition 1.28. □

For groups, the converse also holds.

Theorem 1.34. [1] *A group is k-supernilpotent if and only if it is k-nilpotent.*

This is not true for loops, see Example 2.2.

2. Supernilpotence in loops

In this chapter, we present our results regarding supernilpotent loops. In Section 2.1, we present a simple proof of Proposition 1.33 for loops. In Section 2.2, we characterize 1- and 2-supernilpotent loops and prove some necessary conditions for 3-supernilpotent loops using inner mappings. In Section 2.3 we prove a stronger version of the forward implication in Theorem 1.13. We will use two different approaches to supernilpotence coming from Theorem 1.30 (1) and (3).

In order to use the absorbing polynomials approach, note that in loops all absorbing polynomials can be without loss of generality considered to be absorbing at $(1, ..., 1)$ into 1. This is because a loop polynomial $f(x_1, ..., x_n)$ is absorbing at $\mathbf{a} \in Q$ into $e \in Q$ if and only if the polynomial $f^*(x_1, ..., x_n) = f(x_1a_1, ..., x_na_n)/e$ is absorbing at $(1, ..., 1)$ into 1. Hence, a loop is k-supernilpotent if and only if all these canonical absorbing polynomials of arity $k+1$ are constantly equal to 1. From now on, when we say that a loop polynomial is absorbing, we mean that it is absorbing at $(1, ..., 1)$ into 1.

2.1 Proof of nilpotence of supernilpotent loops

In some cases, the situation can get much clearer, when we interpret the universal-algebraic notion in the context of a particular algebraic structure. Consider Proposition 1.33. The general proof, which is essentially the proof of Proposition 1.27 (8), is rather technical, however, there is a simple proof for groups or loops, which we present in this section.

In the group case, it suffices to observe that the term $[x_1, [x_2, \dots, [x_k, x_{k+1}] \dots]$ is absorbing for every $k \geq 1$ and hence, by Theorem 1.30 (3), constant in every k-supernilpotent group, which implies that the group is k-nilpotent. Using a similar approach, we can prove the proposition also for loops.

Proof of Proposition 1.33 for loops. Let Q be a loop. First we define the elementwise commutator and associator in loops. Let $[x, y] = T_x(y)/y$, $[x, y, z] = L_{x,y}(z)/z$. These are absorbing terms in any loop. Notice that $q \in Z(Q)$ if and only if

$$[q, u] = [u, q] = [q, u, v] = [u, q, v] = [u, v, q] = 1 \text{ for all } u, v \in Q. \tag{2.1}$$

Let us fix a set of variables X. We define *c-terms* over the variable set X recursively: all terms of the form $[x, y]$, $[x, y, z]$, where $x, y, z \in X$ are c-terms of depth 1, and if t is a c-term of depth k, then $[x, t]$, $[t, x]$, $[t, x, y]$, $[x, t, y]$,$[x, y, t]$, where $x, y \in X$, are c-terms of depth $k + 1$. Observe that all c-terms are absorbing.

We prove by induction on k that a loop Q is k-nilpotent if and only if $t(\mathbf{a}) = 1$ for every c-term t of depth k and every tuple $\mathbf{a} \in Q$. For $k = 1$, the condition is equivalent to associativity and commutativity, which is equivalent to Q being an abelian group. Let $k > 1$, then by Proposition 1.9 Q is k-nilpotent if and only if $Q/Z(Q)$ is $(k-1)$-nilpotent, which, by the induction assumption, happens if and only if $t(\mathbf{a}) \in Z(Q)$ for every c-term t of depth $k - 1$ and every tuple $\mathbf{a} \in Q$, which happens if and only if (2.1) holds for $q = t(\mathbf{a})$, which is the same condition as $s(\mathbf{b}) = 1$ for every c-term s of depth k and every tuple $\mathbf{b} \in Q$.

Finally, assume that Q is k-supernilpotent. Every c-term is absorbing, hence every c-term with at least $k+1$ variables is constant. In particular, this applies to every c-term of depth k, hence Q is k-nilpotent. $\square$

2.2 Identities defining supernilpotence

Nilpotence in loops is defined by certain commutator and associator identities (a possible way how to do it can be extracted from the proof in Section 2.1). This motivates a similar approach to supernilpotence. In this section, we derive some identities that are satisfied in supernilpotent loops using the characterization via absorbing polynomials.

Proposition 2.1.

1. *A loop is* 1*-supernilpotent (i.e., abelian) if and only if it is an abelian group.*
2. *A loop is* 2*-supernilpotent if and only if it is a* 2*-nilpotent group.*
3. *In a* 3*-supernilpotent loop Q, for every $x, y, u, v \in Q$ the following is true:*
   - $L_{x,y}$, $R_{x,y}$ *and* $[L_x, R_y]$ *are automorphisms of* Q,
   - $[L_{x,y}, L_{u,v}] = [L_{x,y}, R_{u,v}] = [R_{x,y}, R_{u,v}] = [L_{x,y}, T_u] = [R_{x,y}, T_u] = 1$.

Proof. ($\Leftarrow$) in the first two cases follows from Theorem 1.34.

(1) ($\Rightarrow$) Consider a binary term $t(x, y) = T_x(y)/y$ and a ternary term $l(x, y, z) = L_{x,y}(z)/z$. They are both absorbing, hence constant, therefore a 1-supernilpotent loop needs to be commutative and associative, which implies that it is an abelian group.

(2) ($\Rightarrow$) The absorbing term l from the previous point is constant, hence 2-supernilpotent loops are associative and therefore groups. The rest is by Theorem 1.34.

(3) Observe that for all $x, y \in Q$ the mappings $L_{x,y}$, $R_{x,y}$, T_x and $[L_x, R_y]$ satisfy that $L_{1,y} = L_{x,1} = R_{1,y} = R_{x,1} = T_1 = [L_1, R_y] = [L_x, R_1]$ is the identity mapping and $L_{x,y}(1) = R_{x,y}(1) = T_x(1) = [L_x, R_y](1) = 1$.

For the first claim, let $f_{x,y} : Q \to Q$, $x, y \in Q$ be a bijective mapping and define a term

$$t_f(x, y, u, v) = f_{x,y}(uv)/(f_{x,y}(u)f_{x,y}(v)).$$

If we set $f_{x,y}$ to be $L_{x,y}$, $R_{x,y}$ or $[L_x, R_y]$, respectively, the resulting term will be absorbing and, if Q is 3-supernilpotent, constant, which implies that $f_{x,y}$ is an automorphism.

For the second claim, consider mappings $f_{x,y}, g_{x,y} : Q \to Q$, $x, y \in Q$ and a term

$$t(x, y, u, v, z) = [f_{x,y}, g_{u,v}](z)/z.$$

If we set $(f_{x,y}, g_{u,v})$ to be any of $(L_{x,y}, L_{u,v})$, $(L_{x,y}, R_{u,v})$, $(R_{x,y}, R_{u,v})$, then the resulting term is absorbing and, if Q is 3-supernilpotent, constant. For the pairs $(L_{x,y}, T_u)$, $(R_{x,y}, T_u)$, the claim is proved similarly, we just need to consider mapping g_x instead of $g_{x,y}$ in order to get an absorbing term. $\square$

In the following example, we use the proposition to show that the analogy of Theorem 1.34 for loops does not hold.

Example 2.2. For an odd prime p, consider the set $\mathbb{Z}_{p^2}$ equipped with a binary operation $*$ defined $x * y = x + y + px^2y$, where addition and multiplication are modulo p^2. Then $(\mathbb{Z}_{p^2}, *)$ is a loop of nilpotence class 2, which is supernilpotent of class greater than 2.

Proof. Observe that it is enough to show that $Q = (\mathbb{Z}_{p^2}, *)$ is a non-associative loop of nilpotence class 2. Then since Q is of prime power order, it needs to be supernilpotent (Theorem 1.32), but the supernilpotence class is by Proposition 2.1 (2) greater than 2.

It is straightforward to check that Q is a loop with neutral element 0 and that for all $x, y, z \in Q$

$$(x * y) * z = x * (y * z) \quad \text{iff} \quad pxyz \equiv 0 \pmod{p^2} \quad \text{iff} \quad p|xyz \tag{2.2}$$

(we use that $p \neq 2$). This is clearly not satisfied for example for $(x, y, z) = (1, 1, 1)$, hence Q is not associative. In particular, Q is not 1-nilpotent.

Now we prove that Q is 2-nilpotent. By Proposition 1.9, this is equivalent to proving that $Q/Z(Q)$ is 1-nilpotent, that is, an abelian group. Recall that $Z(Q)$ is a set of elements of Q that commute and associate with all elements of Q. It follows from (2.2) that if $a \in Z(Q)$, then $p|a$ (choose the remaining two elements equal to 1). It easy to see that for all $x, y \in Q$

$$x * y = y * x \quad \text{iff} \quad p(x^2y - y^2x) \equiv 0 \pmod{p^2} \quad \text{iff} \quad p|xy(x - y). \tag{2.3}$$

Combining (2.2) and (2.3), we see that $Z(Q) = \{a \in Q \mid p|a\}$.

To prove that $Q/Z(Q)$ is associative, we need to verify that

$$((x * y) * z) * Z(Q) = (x * (y * z)) * Z(Q),$$

which is equivalent to

$$(x * (y * z))\backslash((x * y) * z) \in Z(Q).$$

Observe that for all $a, b \in Q$, we have $a\backslash b = \frac{b-a}{1+pa^2}$ (the division is modulo p^2 and the denominator is clearly invertible). We compute

$$(x * (y * z))\backslash((x * y) * z) = \frac{2pxyz}{1 + p(x + y + z)^2}$$

and we see that the result is divisible by p, hence belongs to $Z(Q)$.

To prove that $Q/Z(Q)$ is commutative, we verify that

$$(x * y) * Z(Q) = (y * x) * Z(Q),$$

equivalently,

$$(y * x)\backslash(x * y) = \frac{p(x^2y - y^2x)}{1 + p(x + y)^2} \in Z(Q),$$

which again follows from the divisibility by p.

It follows that $Q/Z(Q)$ is an abelian group and hence Q is 2-nilpotent. □

2.3 Connection with the multiplication group

In this section we prove a stronger version of the forward implication in Theorem 1.13 without the finiteness assumption. Using Theorem 1.32, we can reformulate Theorem 1.13 in the following way: *A finite loop Q is supernilpotent if and only if* $\mathrm{Mlt}(Q)$ *is nilpotent.* The following Theorem 2.4 generalizes the forward implication of the theorem for all loops and includes the degrees of supernilpotence, resp. nilpotence.

Firstly, we observe that terms over the multiplication group of a loop can be naturally converted to terms over the corresponding loop. For this transition, we need the following terminology.

Definition 2.3 (m-word)**.** *Let X be a fixed set of variables. Formal expressions of the form*

$$(U^{(1)}_{x_1})^{k_1}\dots(U^{(n)}_{x_n})^{k_n},$$

where $U^{(i)} \in \{L, R\}$, $x_i \in X$ and $k_i \in \{\pm 1\}$, will be called m-words *of length n.*

Observe that every m-word W can be converted naturally into a loop term $W(z)$, $z \in X$, by interpreting $L_{x_i}(z) = x_i z$, $L^{-1}_{x_i}(z) = x_i\backslash z$, $R_{x_i}(z) = z x_i$ and $R^{-1}_{x_i}(z) = z/x_i$ recursively. Every m-word W can also be converted naturally into a mapping $W_{q_1,\dots,q_n} \in \mathrm{Mlt}(Q)$, by replacing x_i by $q_i \in Q$ and interpreting the word as a product in $\mathrm{Mlt}(Q)$.

Indeed, since $\mathrm{Mlt}(Q)$ is generated by left and right translations, for every $h \in \mathrm{Mlt}(Q)$ there is an m-word W and a tuple $\mathbf{q}$ such that $h = W_{\mathbf{q}}$. Moreover, every pair of mappings $h_1, h_2 \in \mathrm{Mlt}(Q)$ can be derived from the same word: if $h_1 = W_{\mathbf{q}}$ and $h_2 = V_{\mathbf{r}}$, then $h_1 = (WV)_{\mathbf{q},1,\dots,1}$ and $h_2 = (WV)_{1,\dots,1,\mathbf{r}}$.

Now we are ready to prove the theorem. The proof shows how to translate Bulatov's term condition for $\mathrm{Mlt}(Q)$ into a term condition for Q.

Theorem 2.4. *Let Q be a k-supernilpotent loop. Then* $\mathrm{Mlt}(Q)$ *is k-nilpotent.*

Proof. Let Q be a k-supernilpotent loop. We will show that $\mathrm{Mlt}(Q)$ is k-supernilpotent by verifying Bulatov's term condition. Then, by Theorem 1.34, $\mathrm{Mlt}(Q)$ is k-nilpotent. Since groups are Mal'tsev, Proposition 1.18 allows us to consider only single elements instead of tuples in the term condition.

Let $t(x_1, \dots, x_{k+1})$ be a group term and $f_1 \dots, f_k$, $g_1, \dots, g_k$, u, $v \in \mathrm{Mlt}(Q)$. Suppose that

$$t(h_1, \dots, h_k, u) = t(h_1, \dots h_k, v)$$

for all $(h_1, \dots, h_k) \in \{f_1, g_1\} \times \cdots \times \{f_k, g_k\} \setminus \{(g_1, \dots, g_k)\}$. We will show that then

$$t(g_1, \dots g_k, u) = t(g_1, \dots, g_k, v).$$

For every $i = 1, \dots, k$, consider an m-word W^i and tuples $\mathbf{a_i}, \mathbf{b_i} \in Q^{l_i}$ such that $f_i = W^i_{\mathbf{a_i}}$ and $g_i = W^i_{\mathbf{b_i}}$. We consider the letters of $W^i_{\mathbf{a_i}}$ ordered in a tuple and denote the tuple by $\mathbf{F_i}$. The elements of $\mathbf{F_i}$ are now from the set $\{L_x, L_x^{-1}, R_x, R_x^{-1} : x \in Q\}$ and their product is equal to f_i when we interpret them as elements of $\mathrm{Mlt}(Q)$. Similarly, we form a tuple $\mathbf{G_i}$ of the letters of $W^i_{\mathbf{b_i}}$. Observe that the tuples $\mathbf{F_i}, \mathbf{G_i} \in \mathrm{Mlt}(Q)^{l_i}$.

In the same way, we express u and v obtaining an m-word W^{k+1} and tuples $\mathbf{c}, \mathbf{d} \in Q^{l_{k+1}}$ such that $u = W^{k+1}_{\mathbf{c}}$ and $v = W^{k+1}_{\mathbf{d}}$. Then we form tuples $\mathbf{U}, \mathbf{V} \in \mathrm{Mlt}(Q)^{l_{k+1}}$ similarly as $\mathbf{F_i}$ and $\mathbf{G_i}$.

We now consider a new group term, $t'(x_1^1, \dots x_1^{l_1}, \dots, x_{k+1}^1, \dots, x_{k+1}^{l_{k+1}})$, which is obtained from t by replacing each occurrence of the variable x_j by the product $x_j^1 \dots x_j^{l_j}$, $j = 1, \dots, k+1$. We observe that

$$\begin{aligned} t'(\mathbf{H_1}, \dots, \mathbf{H_k}, \mathbf{U}) &= t'(\mathbf{H_1}, \dots \mathbf{H_k}, \mathbf{V}), \text{ and hence} \\ t'(\mathbf{H_1}, \dots, \mathbf{H_k}, \mathbf{U})(q) &= t'(\mathbf{H_1}, \dots \mathbf{H_k}, \mathbf{V})(q) \end{aligned}$$

for all $(\mathbf{H_1}, \dots, \mathbf{H_k}) \in \{\mathbf{F_1}, \mathbf{G_1}\} \times \dots \times \{\mathbf{F_k}, \mathbf{G_k}\} \setminus \{(\mathbf{G_1}, \dots, \mathbf{G_k})\}$ and for every $q \in Q$. Since elements of the tuples $\mathbf{F_i}, \mathbf{G_i}, \mathbf{U}, \mathbf{V}$ are from the set $\{L_x, L_x^{-1}, R_x, R_x^{-1} : x \in Q\}$ and t' is a group term, the second set of equalities can be expressed as

$$s(\mathbf{q_1}, \dots, \mathbf{q_k}, \mathbf{c}, q) = s(\mathbf{q_1}, \dots \mathbf{q_k}, \mathbf{d}, q),$$

for all $(\mathbf{q_1}, \dots, \mathbf{q_k}) \in \{\mathbf{a_1}, \mathbf{b_1}\} \times \dots \times \{\mathbf{a_k}, \mathbf{b_k}\} \setminus \{(\mathbf{b_1}, \dots, \mathbf{b_k})\}$ and for every $q \in Q$, where s is a suitable loop term. Since Q is k-supernilpotent, we obtain that for every q

$$\begin{aligned} s(\mathbf{b_1}, \dots, \mathbf{b_k}, \mathbf{c}, q) &= s(\mathbf{b_1}, \dots \mathbf{b_k}, \mathbf{d}, q), \text{which implies} \\ t'(\mathbf{G_1}, \dots, \mathbf{G_k}, \mathbf{U})(q) &= t'(\mathbf{G_1}, \dots \mathbf{G_k}, \mathbf{V})(q) \text{ and hence} \\ t(g_1, \dots g_k, u) &= t(g_1, \dots, g_k, v), \end{aligned}$$

as we wanted to prove. □

Example 2.5. We illustrate the key ideas of the proof on an example. Let us have the following:

- 2-supernilpotent loop Q, $a, b, c \in Q$,
- a group term $t(x_1, x_2, x_3) = x_2 x_3 x_1^{-1}$,
- $f_1 = L_a L_b$, $g_1 = L_b = L_1 L_b$,
- $f_2 = R_c L_a^{-1} = R_c L_a^{-1} R_1^{-1}$, $g_2 = R_b R_a^{-1} = R_b L_1^{-1} R_a^{-1}$,
- $u = R_c^{-1} = R_c^{-1} R_1 R_1$, $v = R_b R_a = R_1^{-1} R_b R_a$.

We have the elements of $\mathrm{Mlt}(Q)$ expressed in m-words, for example, $W^2 = R_{x_1} L_{x_2}^{-1} R_{x_3}^{-1}$, $\mathbf{a_2} = (c, a, 1)$, $\mathbf{b_2} = (b, 1, a)$, $\mathbf{F_2} = (R_c, L_a^{-1}, R_1^{-1})$ and $\mathbf{G_2} = (R_b, L_1^{-1}, R_a^{-1})$.

We have $l_1 = 2$, $l_2 = 3$ and $l_3 = 3$. The term t' will be defined as

$$t'(x_1^1, x_1^2, x_2^1, x_2^2, x_2^3, x_3^1, x_3^2, x_3^3) = x_2^1 x_2^2 x_2^3 x_3^1 x_3^2 x_3^3 (x_1^1 x_1^2)^{-1}.$$

We will illustrate the translation of one equation $t(f_1, f_2, u) = t(f_1, f_2, v)$. The following are equivalent:

$$\begin{aligned} t(f_1, f_2, u) &= t(f_1, f_2, v) \\ t(L_a L_b, R_c L_a^{-1}, R_c^{-1}) &= t(L_a L_b, R_c L_a^{-1}, R_b R_a) \\ t(L_a L_b, R_c L_a^{-1} R_1^{-1}, R_c^{-1} R_1 R_1) &= t(L_a L_b, R_c L_a^{-1} R_1^{-1}, R_1^{-1} R_b R_a) \\ t'(L_a, L_b, R_c, L_a^{-1}, R_1^{-1}, R_c^{-1}, R_1, R_1) &= t'(L_a, L_b, R_c, L_a^{-1}, R_1^{-1}, R_1^{-1}, R_b, R_a) \\ R_c L_a^{-1} R_1^{-1} R_c^{-1} R_1 R_1 L_b^{-1} L_a^{-1} &= R_c L_a^{-1} R_1^{-1} R_1^{-1} R_b R_a L_b^{-1} L_a^{-1} \\ \forall q \in Q : \quad R_c L_a^{-1} R_1^{-1} R_c^{-1} R_1 R_1 L_b^{-1} L_a^{-1}(q) &= R_c L_a^{-1} R_1^{-1} R_1^{-1} R_b R_a L_b^{-1} L_a^{-1}(q) \\ \forall q \in Q : \quad s(a, b, c, a, 1, c, 1, 1, q) &= s(a, b, c, a, 1, 1, b, a, q) \end{aligned}$$

for a loop term

$$s(y_1^1, y_1^2, y_2^1, y_2^2, y_2^3, y_3^1, y_3^2, y_3^3, z) = R_{y_2^1} L_{y_2^2}^{-1} R_{y_2^3}^{-1} R_{y_3^1}^{-1} R_{y_3^2} R_{y_3^3} L_{y_1^2}^{-1} L_{y_1^1}^{-1}(z).$$

If we assume that the equations $t(f_1, f_2, u) = t(f_1, f_2, v)$, $t(g_1, f_2, u) = t(g_1, f_2, v)$, $t(f_1, g_2, u) = t(f_1, g_2, v)$ hold in $\mathrm{Mlt}(Q)$, using this translation for all three equations we derive three loop equations in Q with the term s (same for all equations) and then using 2-supernilpotence of Q, we derive the last equation first in Q and then translate it to $\mathrm{Mlt}(Q)$.

3. Algorithmic testing of supernilpotence

In this chapter we present results from algorithmic testing of supernilpotence of concrete examples of loops. We used the relational description of supernilpotence from Theorem 1.30 (2) to create an algorithm for testing supernilpotence in loops.

Our purpose was to test supernilpotence in finite loops that are known to be supernilpotent of some class. By Theorem 1.32, this involves all nilpotent loops of prime power size. The loops of prime size are either abelian groups or are not nilpotent (since $|Z(Q)|$ divides $|Q|$) and all loops of order 4 are abelian groups, hence the first interesting examples are of order 8 and 9. The primary goal was to test the hypothesis that the reverse implication in Theorem 2.4 also holds or to find examples that violate it.

We used `LOOPS` package [6] for `GAP` [13] to generate the multiplication tables and compute the parameters of multiplication groups of the tested loops, i.e. non-associative nilpotent loops of order 8 and 9.

3.1 Description of the algorithm

The algorithm is based on the relational description of higher commutators by Opršal [9] in terms of Proposition 1.24. The purpose of the algorithm is, for a given positive integer k and a given finite loop Q, to generate the relation

$$\Delta = \Delta(\underbrace{1_Q, \ldots, 1_Q}_{k+1}) = Sg\{c_i^{k+1}(a, b) \mid i < k+1, a, b \in Q\}$$

and check whether there appear forks in the last coordinate. If the algorithm detects a non-trivial fork in Δ, it stops and outputs that Q is not k-supernilpotent. Otherwise, it generates the whole algebra Δ and outputs that Q is k-supernilpotent. The algorithm was implemented in the programming language `C#`. The program is attached to the thesis, see Attachment A.1.

In the generating process, we divide the tuples in Δ into two collections, in the program they are called *Delta* and *Generators*. The tuples in *Delta* correspond to “old” tuples, which were already used for generating another tuples, and the tuples in *Generators* correspond to “new” tuples, which will be used to generate new tuples later on. When we use a tuple for generating, we move the tuple from the collection *Generators* to *Delta*.

Outline of the generating process:

1. Initialize the collection *Delta* consisting of all tuples $c_i^{k+1}(a, a) = (a, \ldots, a)$ (we can do this since a product of two constant tuples is again a constant tuple, and hence we would just obtain duplicates, if we instead put them in the collection Generators) and the collection *Generators* consisting of all tuples $c_i^{k+1}(a, b)$, $a \neq b$.

2. While the collection *Generators* is non-empty and no non-trivial fork was found among all the tuples, perform the following:

   i. Consider the first tuple from the collection *Generators*, denote it by **c**.

   ii. For every tuple **d** in *Delta* compute the tuples $\mathbf{c} \cdot \mathbf{d}$, $\mathbf{d} \cdot \mathbf{c}$ and $\mathbf{c} \cdot \mathbf{c}$. Check if the tuples are not already included in the collection *Delta* or *Generators* and if they do not create a fork with any of the tuples in the collections. If none of these happens, add the tuples to *Generators*. If a fork is found, halt and return `false`.

   iii. Remove the tuple **c** from *Generators* and add it to *Delta*.

3. Return `true` (in this case the generating process ended because of the collection *Generators* being empty and hence we generated whole *Delta*).

Since $\Delta \leq Q^{2^{k+1}}$, it is a finite loop. Therefore, when generating Δ, it is enough to use the operation $\cdot$, since in every finite loop L we have for all $x, y \in L$, $y\backslash x = y \cdot \dots \cdot y \cdot x$, where the number of factors y on the righthand side is equal to the order of L_y in the finite group $\mathrm{Mlt}(L)$ diminished by one and similarly for x/y.

In the implementation of the algorithm, the test of k-supernilpotence of a given loop Q starts with reading a text file corresponding to Q. The text file contains the following information about the loop: size, nilpotence class of $\mathrm{Mlt}(Q)$ (just for comparison with the result) and the multiplication table of Q. An example of a text file used in the program is in the Figure 3.1. The information from the file is stored in the program in a variable *loop* of class *Loop*. After extracting the information from the file, we run the function *TestKSupernilpotence* with parametres k and *loop*, which generates Δ for the given loop and tests the existence of forks in the way outlined above.

```
8
4
1 2 3 4 5 6 7 8
2 1 4 3 6 5 8 7
3 4 5 6 7 8 1 2
4 3 6 5 8 7 2 1
5 6 7 8 1 2 3 4
6 5 8 7 2 1 4 3
7 8 1 2 3 4 6 5
8 7 2 1 4 3 5 6
```

Figure 3.1: Text file for 8-element nilpotent loop no. 1

3.2 Data representation

So far we referred to collections *Delta* and *Generators* without specifying any representation of the tuples in the memory. However, this is a significant aspect of the algorithm implementation, since we need to be able to check existence of duplicates and forks effectively. In our case, this is extremely important, since we are not aware of any tighter upper bound on the size of the generated relation Δ than the naive bound $|Q|^{2^{k+1}}$. After testing the program with a straightforward representation of the collections as *List*

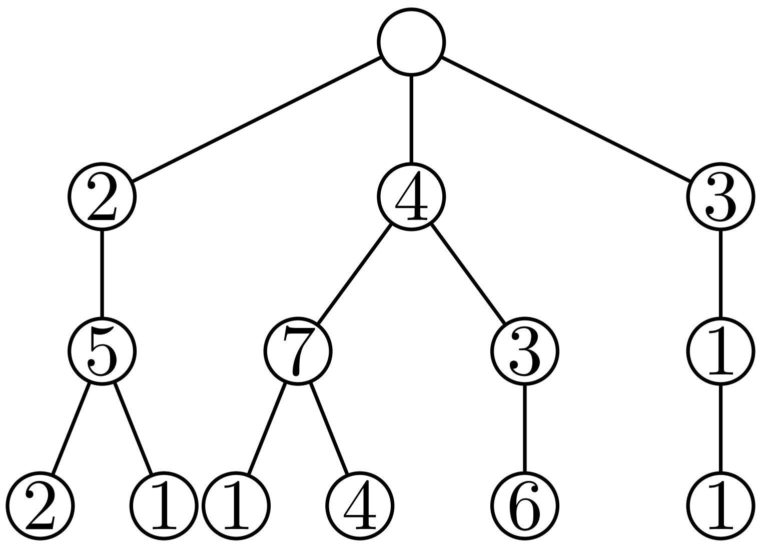

Figure 3.2: A tree of collection $\{(2,5,2),(2,5,1),(4,7,1),(4,7,4),(4,3,6),(3,1,1)\}$

classes, it turned out that the running time in this case is not satisfactory. Therefore, we switched to a representation of the collections in a tree structure.

The idea is the following: Every collection corresponds to a rooted tree. Vertices of the tree will be called *nodes*, every node except for the root has an assigned value from Q (in the implementation $Q = \{1, 2, \ldots |Q|\}$ and the root has a value $> |Q|$ to be distinguished) and there are no nodes with the same parent and the same value. We say that a root is on level 0 and if a node is on a level l, then all of its children are said to be on level $l + 1$. All leaves of the tree are on the same level and this level is equal to the length of the tuples in the collection. An example of a tree of a collection is given in Figure 3.2.

To write it down formally, every tuple $\mathbf{c}$ in the collection corresponds to exactly one path on the way from the root to a leaf, that is, c_i is equal to the value of the node on the level i on the path. Therefore we could define the tree of the collection of tuples C as a rooted tree satisfying the following conditions:

- the values of children of the root are precisely the values in the set
$$\{c_1 : \mathbf{c} \in C\},$$
- if the root $= N_0, N_1, \ldots N_m$ is a path from the root to a non-leaf node N_m, and the value of N_i is v_i, then the values of children of N_m are precisely the elements of
$$\{c_{m+1} : \exists \mathbf{c} \in C \text{ such that } c_i = v_i, i = 1, \ldots, m\}.$$

In the algorithm implementation, for a given collection, we represented its tree using a class *Node*. An instance of this class contains information about the parent node, the list of the children nodes (the list is not ordered by values) and the value of the node. A collection is then represented by its root, which is an instance of this class.

Consider now a collection containing tuples of length m and its tree. Suppose we want to add a tuple $\mathbf{c}$ to the collection. We start in the root and look for its child with a value

c_1. If we find such, we move to this node. In general, if we are at a node N on level $l < m$, we look for a child of a value c_{l+1} and if we succeed, we move to it. Otherwise, we create a node with this value as a child of the node N and move to it (from this moment on we of course create new nodes in every step without the search). Observe that this process can be naturally modified to or even performed together with the check for duplicates and forks.

In a similar manner we remove a given tuple $\mathbf{c}$ from the tree. If we are given the corresponding leaf in the tree, we follow the way from the leaf to the root through the nodes with values c_j, where j is the level of the node. On the way, we remove the leaf and all following nodes that have just one child (which corresponds to a component of the tuple), stopping at the first node with more children.

The tree representation of the collection allows us to perform a check for duplicates and forks (optionally together with addition to the tree) in time $O(|Q| \cdot 2^{k+1})$, when testing k-supernilpotence of a loop Q. This is because if we consider a tuple $\mathbf{c}$ of length 2^{k+1} and a tree of the collection, where we want to perform the check, then for every $j = 1, \dots, 2^{k+1}$, we search for a child of a corresponding node on level j with value c_{j+1}. Since there are $|Q|$ possible values of children, we perform $O(|Q| \cdot 2^{k+1})$ steps. Compared to the list representation of the collection, where adding a tuple is straightforward, but the check for duplicates and forks involves comparing with every tuple in the collection individually, this approach significantly reduces the running time of the program. In the list representation, such check takes $O(2^{k+1}s)$, where s is a size of a collection, which grows rapidly and for which we do not know a better upper bound than $|Q|^{2^{k+1}}$. However, as is discussed in the Section 3.4, it turns out that even with this representation, our program is not fast enough.

3.3 Results

Since we know from Proposition 2.1 that 2-supernilpotent loops are precisely 2-nilpotent groups, 2-supernilpotence does not bring anything new in loops. Therefore, we focused on testing 3-supernilpotence, where the relation Δ consists of tuples of length 16.

According to Theorem 2.4, no loop with a multiplication group of nilpotence class greater than 3 is 3-supernilpotent. Therefore, we ran the algorithm on the non-associative 8-element nilpotent loops whose multiplication group has nilpotence class 3 and all non-associative 9-element nilpotent loops (since they all have multiplication group of nilpotence class 3). We also used non-associative 8-element loops whose multiplication group has nilpotence class 4, the cyclic groups of orders 4 and 8 and the quaternion group to test the program. All tests were run on a standard personal computer.

3.3.1 8-element loops

All non-associative 8-element nilpotent loops are of nilpotence class 2, since the size of the center divides the size of the loop and all loops of order ≤ 4 are abelian groups. According to data from `LOOPS` package for `GAP` [6], their multiplication groups are either of size 64 or 32 and the nilpotence class of the groups is either 3 or 4. We tested 3-supernilpotence in those with multiplication group of nilpotence class 3.

The results are summarized in Table 3.1, which includes the number of the loop in `GAP` (that is, the loop with number n is listed in `GAP` as `NilpotentLoop(8,n)`), nilpotence class of its multiplication group, size of its multiplication group, the output from the 3-supernilpotence test and running time. We can see that out of 62 tested 8-element loops, there are 34 that the program proved to have class of supernilpotence greater than 3, for the rest, the computation did not finish in the given time. In both of these categories we can found loops with multiplication groups of both sizes (64 and 32) with no evident pattern.

The given computation time varies among the loops. We started testing the loops with a time limit of few hours and after running some tests on 8- and 9-element loops (see first rows of Table 3.1 and Table 3.2) we observed that the computation either finished in about a minute, or the time limit did not suffice. Therefore, we changed the time limit to 10 minutes.

number	**nilp. cl.** $\mathrm{Mlt}(Q)$	$\lvert\mathrm{Mlt}(Q)\rvert$	**3-supernilp.**	**running time**
27	3	32	**NO**	73797 ms
42	3	32	?	3 hrs
48	3	32	**NO**	75259 ms
58	3	32	**NO**	73581 ms
73	3	32	**NO**	75809 ms
75	3	32	?	2 hrs
79	3	64	**NO**	73797 ms
80	3	64	**NO**	70767 ms
81	3	64	?	10 min
82	3	64	?	10 min
83	3	64	**NO**	62379 ms
84	3	64	?	10 min
85	3	32	?	10 min
86	3	64	**NO**	64218 ms
87	3	64	?	10 min
88	3	64	**NO**	75757 ms
89	3	64	**NO**	70813 ms
90	3	64	?	10 min
91	3	32	?	10 min
92	3	64	**NO**	65362 ms
93	3	64	?	10 min
94	3	64	**NO**	67078 ms
95	3	64	**NO**	73878 ms
96	3	64	?	10 min
97	3	64	**NO**	75967 ms
98	3	64	?	10 min
99	3	64	?	10 min
100	3	64	**NO**	72434 ms
101	3	64	?	10 min
102	3	64	**NO**	75128 ms

number	**nilp. cl.** Mlt(Q)	$\lvert$Mlt(Q)$\rvert$	**3-supernilp.**	**running time**
103	3	64	**NO**	73318 ms
104	3	32	**?**	10 min
105	3	64	**?**	10 min
106	3	64	**NO**	73160 ms
107	3	64	**?**	10 min
108	3	64	**NO**	70051 ms
109	3	64	**NO**	63173 ms
110	3	64	**?**	10 min
111	3	64	**NO**	61275 ms
112	3	64	**NO**	62917 ms
113	3	64	**NO**	62300 ms
114	3	64	**?**	10 min
115	3	32	**?**	10 min
116	3	64	**NO**	71378 ms
117	3	64	**?**	10 min
118	3	64	**?**	10 min
119	3	64	**NO**	67014 ms
120	3	64	**NO**	70587 ms
121	3	64	**NO**	59952 ms
122	3	32	**?**	10 min
123	3	64	**?**	10 min
124	3	64	**NO**	63441 ms
125	3	64	**NO**	72973 ms
126	3	64	**NO**	68741 ms
127	3	64	**?**	10 min
128	3	64	**NO**	72573 ms
129	3	64	**NO**	66641 ms
130	3	64	**NO**	67719 ms
131	3	64	**?**	10 min
132	3	64	**NO**	67114 ms
133	3	32	**?**	10 min
134	3	64	**?**	10 min

Table 3.1: Results of testing 8-element nilpotent loops

3.3.2 9-element loops

All relevant 9-element loops are of nilpotence class 2, since they are non-associative and nilpotent, hence their center is of size 3 and all 3-element loops are already abelian groups. According to information from the `LOOPS` package [6], each of them has multiplication group of order 81 with nilpotence class 3, so they all could be 3-supernilpotent. We therefore tested all these loops for 3-supernilpotence.

The results are summarized in Table 3.2 (the loop with number n is the loop listed in `GAP` as `NilpotentLoop(9,n)`). Our computation did not finish for any of the tested

number	**nilp. cl.** Mlt(Q)	$\lvert \mathrm{Mlt}(Q) \rvert$	**3-supernilp.**	**running time**
1	3	81	?	3 hrs
2	3	81	?	3 hrs
3	3	81	?	3 hrs
4	3	81	?	3 hrs
5	3	81	?	3,5 hrs
6	3	81	?	3 hrs
7	3	81	?	2 hrs
8	3	81	?	3 hrs

Table 3.2: Results of testing 9-element nilpotent loops

9-element loops in a given time of at least 2 hours. The computation time for individual loops varies only due to time options at the time of the computation.

3.3.3 Groups

In order to compare, we tested k-supernilpotence in the cyclic groups $\mathbb{Z}_4$ and $\mathbb{Z}_8$, and the quaternion group Q_8 (a smallest non-abelian nilpotent group) for $k = 2, 3, 4$. The results are summarized in Table 3.3. The computation time "$> x$ hrs" means that in x hours, the computation did not finish and we stopped it. These times are arbitrary and differ only due to time options at the time of the test.

group	**run. time for** $k = 2$	**run. time for** $k = 3$	**run. time for** $k = 4$
$\mathbb{Z}_4$	39 ms	908 ms	22225 ms
$\mathbb{Z}_8$	7630 ms	18 min	$>$ 4 hrs
Q_8	10 min	$>$ 7 hrs	did not run

Table 3.3: Results of testing groups

3.4 Conclusions

Counterexamples to reverse implication in Theorem 2.4: We found a number of 8-element nilpotent loops which have a multiplication group of nilpotence class 3 and are not 3-supernilpotent. Therefore, the reverse implication to Theorem 2.4 is not true. Moreover, this provides numerous examples that can be studied to understand the reasons that lead to existence of nontrivial forks in Δ relation and therefore prevent loop from being 3-supernilpotent.

Computation time: Even after the modifications of data representation that significantly reduce running time, we did not manage to obtain a result confirming 3-supernilpotence of any of these loops in 10 minutes of computation (and for some of the loops in more than 3 hours of computation). The results obtained from testing supernilpotence in groups contribute to a belief that a positive result for a 3-supernilpotent non-associative loop would take a long time, since the computation for the group Q_8 did not finish in more than 7 hours.

Does ? mean YES?: We can observe that in every loop, which was proved by the computation not to be 3-supernilpotent, it took less than 90 seconds to get the result. This leads to a conjecture that the remaining loops are 3-supernilpotent, especially for the 8-element case. Also this might indicate that if there is a nontrivial fork in Δ relation, there exist many of them and hence there might be a way to decide faster, whether a loop is k-supernilpotent or not.

Testing for $k > 3$: Naturally, testing k-supernilpotence for $k > 3$ in loops does not seem to be promising at the moment. Clearly, positive answers cannot be achieved in a reasonable time. Furthermore, obtaining a negative answer does not seem realistic at the moment as well, since we did not get any output from the tested loops for $k = 4$ in 20 minutes (we tested more than a half of the loops, which were proved not to be 3-supernilpotent) and from some of the loops in more than 7 hours. Therefore, there are probably needed significant improvements in the algorithm in order to enable its wider use in testing supernilpotence.

Conclusion

The purpose of the thesis was to study properties of higher commutators in loops and in particular, to study supernilpotent loops. Our original plan was to build on results in group theory, since it is known that in groups supernilpotence and nilpotence coincide, and to search for a description of the higher commutator (at least in special cases) using a similar approach to the one that was used to describe the binary commutator in loops in [12]. Moreover, we focused on the theorem of Wright from [14] that shows that, for a finite loop, supernilpotence is equivalent to nilpotence of its multiplication group. In particular, we tried to find a syntactic proof, possibly without the finiteness assumption.

However, a closer study of the group case showed that the proof of the key theorem which states that k-nilpotent groups are k-supernilpotent [1, Theorem 6.8] is based on a detailed syntactic analysis of polynomial functions on groups and rewriting them using the group commutators. Therefore, it did not seem feasible to try to modify the proof in order to obtain some non-trivial results about loops, where the analysis would also require considering associators.

Regarding the description of the higher commutator, we observed that the general case is quite hard to approach and hence, we focused on the special case of supernilpotence. In Proposition 2.1 we characterize 1- and 2-supernilpotent loops and give a number of necessary conditions for 3-supernilpotent loops. We were inspired by the mentioned description of the binary commutator of two normal subloops from [12].

The most successful was the study of Wright's theorem. In Section 2.3 we present a syntactic proof of the forward implication without the finiteness assumption in a stronger version, since we showed that a k-supernilpotent loop has a k-nilpotent multiplication group.

Nonetheless, it turned out that it is difficult to find non-trivial sufficient conditions for k-supernilpotence even for the $k = 3$. This motivated us to study 3-supernilpotence on concrete examples. In order to do this, we created and implemented an algorithm for testing k-supernilpotence in finite loops based on Opršal's relational description of supernilpotence. The results of the tests showed, among other things, that k-nilpotence of a multiplication group does not imply k-supernilpotence of the loop, even in the cases when we know that the loop is supernilpotent of some class. The results of this algorithmic testing provide a number of examples that can be used in further research.

The thesis still leaves a lot of questions regarding higher commutators and supernilpotence in loops open. It would be desirable, for example, to characterize 3-supernilpotent loops (and possibly k-supernilpotent loops in general) in a loop-theoretical fashion, or to examine the remaining implication of Wright's theorem using universal-algebraic tools. Also, the actual version of the algorithm from Chapter 3 seems to be unable to confirm 3-supernilpotence (and of course also supernilpotence of a greater class) in a reasonable time limit in non-associative loops, even if they are of small orders. However, the running time of the test in non-3-supernilpotent loops indicates that there might be a way to decide faster, whether a loop is 3-supernilpotent or not. These questions could be a subject of further studies.

Bibliography

[1] E. Aichinger and J. Ecker. Every $(k+1)$-affine complete nilpotent group of class k is affine complete. *Internat. J. Algebra Comput.*, 16(2):259–274, 2006.

[2] E. Aichinger and N. Mudrinski. Some applications of higher commutators in Mal'cev algebras. *Algebra Universalis*, 63(4):367–403, 2010.

[3] R. H. Bruck. Contributions to the theory of loops. *Trans. Amer. Math. Soc.*, 60:245–354, 1946.

[4] A. Bulatov. On the number of finite Mal'tsev algebras. In *Contributions to general algebra, 13 (Velké Karlovice, 1999/Dresden, 2000)*, pages 41–54. Heyn, Klagenfurt, 2001.

[5] R. Freese and R. McKenzie. *Commutator theory for congruence modular varieties*, volume 125 of *London Mathematical Society Lecture Note Series.* Cambridge University Press, Cambridge, 1987.

[6] G. P. Nagy and P. Vojtěchovský. `LOOPS`: Computing with quasigroups and loops in `GAP`. `https://www.gap-system.org/Packages/loops.html`, 2018. Version 3.4.1.

[7] H.-P. Gumm. Algebras in permutable varieties: geometrical properties of affine algebras. *Algebra Universalis*, 9(1):8–34, 1979.

[8] M. Niemenmaa and M. Rytty. Centrally nilpotent finite loops. *Quasigroups Related Systems*, 19(1):123–132, 2011.

[9] J. Opršal. A relational description of higher commutators in Mal'cev varieties. *Algebra Universalis*, 76(3):367–383, 2016.

[10] J. D. H. Smith. *Mal'cev varieties.* Lecture Notes in Mathematics, Vol. 554. Springer-Verlag, Berlin-New York, 1976.

[11] D. Stanovský and P. Vojtěchovský. Abelian extensions and solvable loops. *Results Math.*, 66(3-4):367–384, 2014.

[12] D. Stanovský and P. Vojtěchovský. Commutator theory for loops. *J. Algebra*, 399:290–322, 2014.

[13] The GAP Group. `GAP`. `http://www.gap-system.org`, 2020. Version 4.11.0.

[14] C. R. B. Wright. On the multiplication group of a loop. *Illinois J. Math.*, 13:660–673, 1969.

A. Attachments

A.1 Program Algorithmic testing of supernilpotence

A compressed folder with a Visual Studio project Algorithmic testing of supernilpotence, which contains the program from Chapter 3.