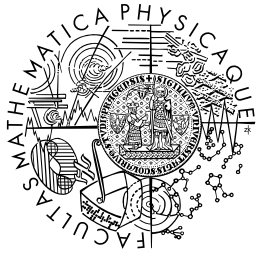


MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Jan Frnka

Kryptografie na copánkových grupách

Katedra algebry

Vedoucí bakalářské práce: Mgr. Adolf Středa

Studijní program: Matematika

Studijní obor: Matematika pro informační technologie

Praha 2024

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ....

Podpis autora

Rád bych poděkoval především svému vedoucímu práce Mgr. Ádovi Středovi za pročtení všech verzí, cenné připomínky, a především pak za spoustu času, který mi věnoval v posledních dnech před odevzdáním. Také děkuji všem čtenářům za zájem a doufám, že Vám práce něco přinese.

Název práce: Kryptografie na copánkových grupách

Autor: Jan Frnka

Katedra: Katedra algebry

Vedoucí bakalářské práce: Mgr. Adolf Středa, Katedra algebry

Abstrakt: Copánkové grupy obsahují některé problémy, které umožňují vytvoření trapdoor funkcí pro účely asymetrické kryptografie. Konkrétně conjugacy problem ukazoval potenciál pro tyto účely a vzniklo několik schémat na něm založených. Bohužel se záhy ukázalo, že instance tohoto problému používané v návrzích schémat jsou zranitelné vůči útokům. Cílem této práce bude formálně popsat copánkové grupy a vybudovat teorii pro popis tohoto problému, vybraných odvozených kryptosystémů a útoků na tyto kryptosystémy. V závěru práce pak nahlédneme na další potenciální problémy, které by mohly sloužit k vybudování nového asymetrického kryptosystému.

Klíčová slova: copánková grupa, pozitivní copánky, fundamentální copánek, permutační copánky, konjugace, asymetrická kryptografie

Title: Braid Group Cryptography

Author: Jan Frnka

Department: Department of Algebra

Supervisor: Mgr. Adolf Středa, Department of Algebra

Abstract: Braid groups involve certain problems that enable the construction of trapdoor functions for the purposes of asymmetric cryptography. Specifically, the conjugacy problem has shown potential in this direction, leading to the development of several schemes. However, it was soon revealed that instances of this problem used in designed schemes are vulnerable to attacks. The aim of this thesis is to formally describe braid groups and construct a theoretical framework to study this problem, selected derived cryptosystems, and attacks on these cryptosystems. In the conclusion, we will explore further potential problems that could be utilized to construct a new asymmetric cryptosystem.

Keywords: braid group, positive braids, fundamental braid, permutation braids, conjugation, asymmetric cryptography

Obsah

Úvod

Myšlenka, že zapletené provázky stojí za zájem matematiků, se poprvé objevila u Carla Friedricha Gausse na začátku 18. století. Krom běžného copu ze tří pramenů, který známe ze zaplétání vlasů, se dle náčrtů v jeho poznámkách začal zabývat i obecnějšími copánky. V průběhu let se copánky objevily v několika pracích, ale až v roce 1925 Emil Artin v (Artin, 1925) přišel s formální definicí copánků na n provázcích a dokázal, že odpovídá geometrické intuici.

Při dalším výzkumu copánkových grup se podařilo objevit několik potenciálních „trapdoor“ funkcí, tedy funkcí, které je snadné spočítat, ale obtížné invertovat bez znalosti určité další informace. Kolem roku 2000 vzniklo několik článků navrhujících asymetrická kryptografická schémata pracující na copánkových grupách, z nichž většina využívala složitost takzvaného *conjugacy problem*. Poměrně rychle se však ukázalo, že tento problém není tak obtížný, jak se předpokládalo, a zhruba od roku 2008 se z hlediska kryptografie věnuje copánkovým grupám minimální pozornost.

V této práci se po vybudování základní teorie zaměříme především na *conjugacy problem*, na němž je založená takřka veškerá kryptografie na copánkových grupách. V druhé kapitole tak důkladně popíšeme původní řešení tohoto problému, které navrhl (Garside, 1969), a nabídneme elementárnější důkaz jeho platnosti. Následně se podíváme i na účinnější řešení navržené (Gebhardt, 2003) a na experimentální výsledky demonstrující jeho efektivitu.

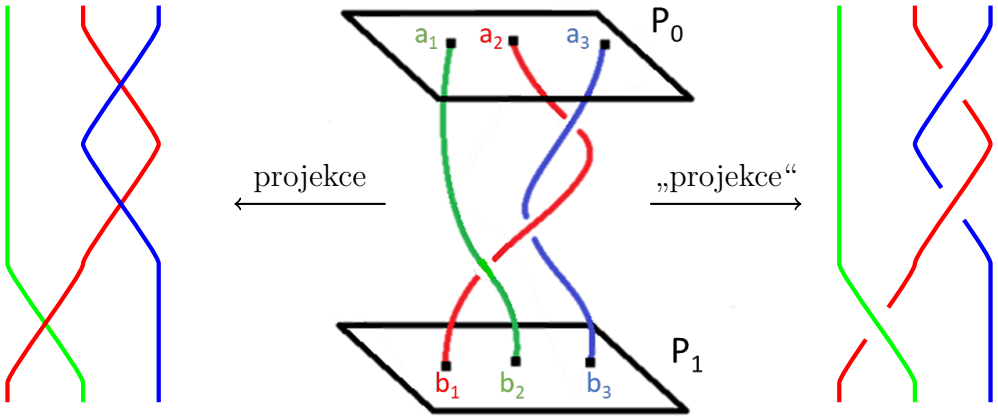
V poslední kapitole si představíme některé kryptosystémy na copánkových grupách a rozmyslíme si jejich bezpečnost, tedy efektivitu dosud známých útoků. Také se krátce zaměříme na potenciální „trapdoor“ funkce, které nejsou založené na *conjugacy problem* a navrhneme nový možný směr, kterým by se kryptografie na copánkových grupách mohla dále ubírat.

1. Copánkové grupy

V této práci budeme pracovat s takzvanou *artinovou prezentací* copánkových grup, kterou navrhl Emil Artin v (Artin, 1925). Než se však k jeho definici dostaneme, představíme geometrickou interpretaci copánků, která více odpovídá intuici.

Uvažujme rovnoběžné roviny P_0,P_1 v $\mathbb{R}^3$ a po dvou různé body $a_1, \ldots, a_n \in P_0$, $b_1, \ldots, b_n \in P_1$. Copánek je pak tvořen n provázky, kde i-tý provázek začíná v bodě a_i a končí v bodě $b_{\pi(i)}$ pro nějakou permutaci $\pi \in S_n$. Provázky se mohou zakrucovat, ale nesmí tvořit uzly, ani se žádný provázek nemůže křížit se sebou samým. Příklad takového copánku můžeme vidět na obrázku 1.1 uprostřed. Pokud jde mezi dvěma copánky přecházet „taháním za provázky“, pak tyto copánky prohlásíme za stejné.

Tento popis copánků nejlépe odpovídá intuici, pro snazší reprezentaci však budeme pracovat s kolmou projekcí do $\mathbb{R}^2$ (takovou, že se žádné dva různé body na P_0 a P_1 nezobrazí na stejný bod) viz obrázek 1.1 vlevo.



Obrázek 1.1: Příklad copánku a jeho projekce do $\mathbb{R}^2$

Projekce přirozeně vede k překryvu provázků, proto musíme rozlišovat, který z provázků je nahoře (dál od projekční roviny), viz obrázek 1.1 vpravo. To vede k označení *pozitivního* a *negativního* křížení, kde křížení nazveme pozitivním, pokud je levý provázek nad pravým. Na obrázku 1.1 vpravo jsou obě křížení modrého a červeného provázku negativní a křížení zeleného a červeného provázku je pozitivní. Snadno si rozmyslíme, že díky této terminologii jsme ve dvou dimenzích schopni zachytit všechny copánky. Přirozeně je můžeme popisovat právě pomocí křížení, což nás přivádí k formální definici copánkové grupy.

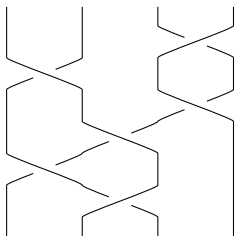
Definice 1. *Pro $n \in \mathbb{N}$: $n \geq 2$ definujeme n-tou copánkovou grupu B_n následovně*

$$B_n = \left\langle \sigma_1, \sigma_2, \ldots, \sigma_{n-1} \;\middle|\; \begin{array}{ll} \sigma_i\sigma_j = \sigma_j\sigma_i & |i-j| > 1 \\ \sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1} & i \leq n-2 \end{array} \right\rangle.$$

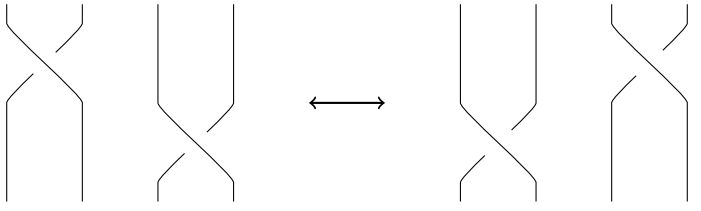
Poznámka. V definici uvažujeme volnou grupu F na $X = \{\sigma_1, \ldots, \sigma_{n-1}\}$, tedy grupu posloupností prvků z $X \cup X^{-1} \cup \{\varepsilon\}$ $(= \{\sigma_1, \ldots, \sigma_{n-1}, \sigma_1^{-1}, \ldots, \sigma_{n-1}^{-1}, \varepsilon\}$, kde $\sigma_i\sigma_i^{-1} = \varepsilon$ (operací na F je skládání posloupností a ε je jednotka). Dále máme normální podgrupu $H \subseteq F$ generovanou množinou $\{\sigma_i\sigma_j\sigma_i^{-1}\sigma_j^{-1} \mid |i-j| > 1\}$ $\cup \{\sigma_i\sigma_{i+1}\sigma_i\sigma_{i+1}^{-1}\sigma_i^{-1}\sigma_{i+1}^{-1} \mid i \leq n-2\}$. Pak $B_n = {}^{F}\!/_{H}$, tj. copánky jsou třídy ekvivalence určené danými rovnostmi.

Pro nás bude postačující následující interpretace definice. Copánky jsou posloupnosti generátorů a jejich inverzů, tyto posloupnosti budeme nazývat *slovy* a délku slova b budeme značit $|b|$. Ve slovech můžeme libovolně nahrazovat $\sigma_i\sigma_j$ za $\sigma_j\sigma_i$ pro $|i-j| > 1$ a $\sigma_i\sigma_{i+1}\sigma_i$ za $\sigma_{i+1}\sigma_i\sigma_{i+1}$ pro $i \leq n-2$ a stále budeme mít ten samý copánek. Tyto dva typy nahrazení indukují relace ekvivalence, ke kterým se budeme nadále odkazovat jako k *relacím*. Jednotka ε odpovídá prázdné posloupnosti a operací je skládání posloupností za sebe. Inverz k $b \in B_n$: $b = \sigma_{i_1}^{m_1}\sigma_{i_2}^{m_2}\cdots\sigma_{i_k}^{m_k}$, kde $m_i \in \{1, -1\}$ a σ_i^1 ztotožňujeme s σ_i, je pak $b^{-1} = \sigma_{i_k}^{-m_k}\sigma_{i_{k-1}}^{-m_{k-1}}\cdots\sigma_{i_1}^{-m_1}$. Nyní si rozmyslíme, jak definice odpovídá geometrické interpretaci — (Artin, 1925) dokonce ukázal, že oba pohledy popisují tentýž objekt, tudíž je mezi nimi možné přecházet.

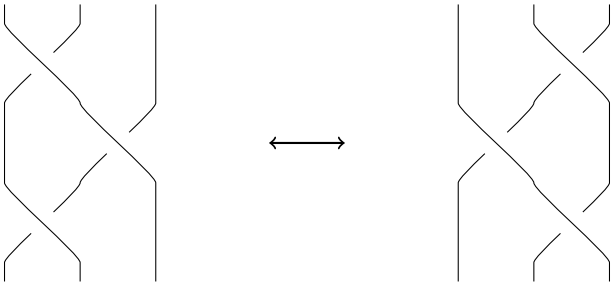
$n \in \mathbb{N}$ je počet provázků, σ_i odpovídá pozitivnímu křížení provázku na i-té pozici s provázkem na $(i+1)$-ní pozici a σ_i^{-1} je negativní křížení. Na obrázcích budeme copánky zakreslovat shora dolů – příklad copánku a slova, které jej popisuje, máme na obrázku 1.2. Skládání je pak navázání dvou copánků na sebe a inverz odpovídá rozmotávání copánku odspodu. Relace pak odpovídají obrázkům 1.3, 1.4.



Obrázek 1.2: copánek $b = \sigma_3^{-1}\sigma_1\sigma_3\sigma_2\sigma_1\sigma_2^{-1}$



Obrázek 1.3: Relace: $\sigma_i\sigma_j = \sigma_j\sigma_i \quad \forall i, j : |i-j| > 1$



Obrázek 1.4: Relace: $\sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1} \quad \forall i \leq n-2$

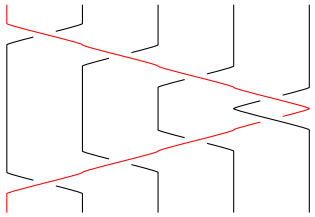
Nyní si ukážeme, jak můžeme aplikací relací upravit copánek na obrázku 1.2.

Příklad.

$$b = \sigma_3^{-1}\sigma_1\sigma_3\sigma_2\sigma_1\sigma_2^{-1} \overset{(1)}{=} \sigma_3^{-1}\sigma_3\sigma_1\sigma_2\sigma_1\sigma_2^{-1} \overset{(2)}{=} \sigma_1\sigma_2\sigma_1\sigma_2^{-1} \overset{(3)}{=} \sigma_2\sigma_1\sigma_2\sigma_2^{-1} \overset{(4)}{=} \sigma_2\sigma_1$$

Kde (1) platí, jelikož $\sigma_1\sigma_3 = \sigma_3\sigma_1$, (2) máme díky $\sigma_3^{-1}\sigma_3 = \varepsilon$, pro (3) využijeme $\sigma_1\sigma_2\sigma_1 = \sigma_2\sigma_1\sigma_2$, a nakonec (4) platí, protože $\sigma_2^{-1}\sigma_2 = \varepsilon$.

Z příkladu vidíme, že copánek b z obrázku 1.2 lze zapsat jako $b = \sigma_2\sigma_1$. Patří tak do užitečné třídy copánků, kterou zavedeme v následující definici.

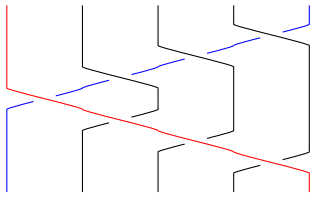


Obrázek 1.5: Lze popsat jako křížení prvního a pátého provázku

Definice 2. *Slovo bez negativních křížení nazveme* pozitivním slovem. *Copánky, které lze popsat pozitivním slovem, nazýváme* pozitivní copánky. *Množinu pozitivních copánků značíme* B_n^+.

1.1 Normální forma

Je zjevné, že v copánkové grupě můžeme díky relacím jeden copánek popsat mnoha různými slovy. Pochopitelně tak vyvstává otázka, jaká varianta zápisu je nejvhodnější.



Obrázek 1.6: Lze popsat jako permutaci $(1\ 5) \in S_5$

Intuitivně se zdá rozumné uvažovat nejkratší možné slovo, i tak však může zbývat mnoho různých způsobů zápisu. Navíc nás může napadnout, že dosažením určité struktury nám pro uložení může stačit výrazně méně místa (viz příklady na obrázcích 1.5, 1.6). Ještě užitečnější než prostorová efektivita je ale jednoznačnost zápisu, která nám mimo jiné umožní porovnat, zda dvě slova popisují ten samý copánek.

V této práci zvolíme jako jednotný zápis takzvanou *normální formu* copánku. K jejímu zavedení potřebujeme zavést několik dalších pojmů počínaje fundamentálním copánkem.

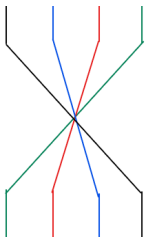
1.1.1 Fundamentální copánek

Definice 3. *Pro B_n definujeme* fundamentální copánek *jako copánek*

$$\Delta_n = (\sigma_1 \cdots \sigma_{n-1})(\sigma_1 \cdots \sigma_{n-2}) \cdots (\sigma_1\sigma_2)(\sigma_1).$$

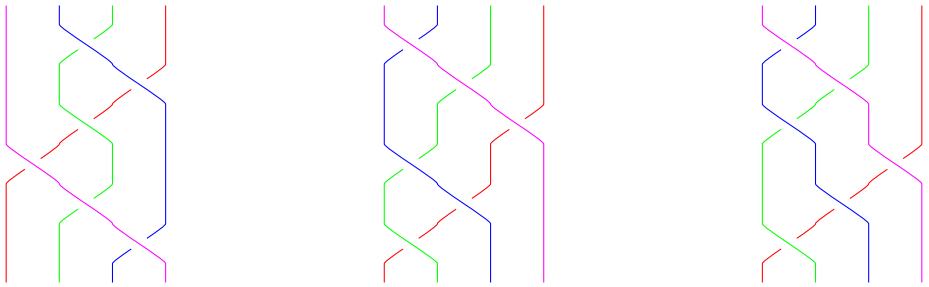
Poznámka. Alternativně můžeme fundamentální copánek definovat induktivně: $\Delta_n = (\sigma_1 \cdots \sigma_{n-1})\Delta_{n-1}$ a $\Delta_2 = \sigma_1$. Obě definice jsou ekvivalentní.

Geometrická interpretace fundamentálního copánku je na obrázku 1.7 a zakreslení některých slov, které jej popisují, máme na obrázku 1.8 (definující slovo vidíme uprostřed). Můžeme učinit ryze geometrické pozorování, které nám přiblíží intuitivní náhled na fundamentální copánek.



Obrázek 1.7: Δ_4

Pozorování. Při nakreslení pozitivních slov popisujících Δ_n získáme copánek, kde se každý pár provázků kříží právě jednou. Proto máme přesně $n(n-1)/2$ křížení, což odpovídá počtu σ_i v definujícím slově.



Obrázek 1.8: Δ_4

Nyní se vrátíme k algebraické interpretaci fundamentálního copánku a ukážeme si, kterými dalšími slovy jej můžeme popsat.

Lemma 1. *Pro $n \in \mathbb{N}$ lze fundamentální copánek zapsat slovy*

$$\Delta_n = (\sigma_1)(\sigma_2\sigma_1) \cdots (\sigma_{n-1} \cdots \sigma_1) \tag{1.1}$$
$$= (\sigma_{n-1})(\sigma_{n-2}\sigma_{n-1}) \cdots (\sigma_1 \cdots \sigma_{n-1}). \tag{1.2}$$

Dále pro $n \geq 4$ a $1 < i < n-1$ jej můžeme zapsat dalšími dvěma slovy

$$\Delta_n = (\sigma_1 \cdots \sigma_{n-1}) \cdots (\sigma_1 \cdots \sigma_{i+1})(\sigma_1)(\sigma_2\sigma_1) \cdots (\sigma_i \cdots \sigma_1) \tag{1.3}$$
$$= (\sigma_i)(\sigma_{i+1}\sigma_i) \cdots (\sigma_{n-1} \cdots \sigma_i)(\sigma_{i-1} \cdots \sigma_{n-1}) \cdots (\sigma_1 \cdots \sigma_{n-1}). \tag{1.4}$$

Důkaz. Nejdřív zopakujeme obě relace, které můžeme využívat pro úpravu slov,

$$\sigma_i\sigma_j = \sigma_j\sigma_i, \qquad \forall i,j : |i-j| > 1 \tag{1.5}$$
$$\sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}. \qquad \forall i \leq n-2 \tag{1.6}$$

Tyto relace nám dávají užitečný vztah pro $1 < j \leq n-1$

$$\sigma_j(\sigma_1\sigma_2 \cdots \sigma_{n-1}) \overset{(1.5)}{=} \sigma_1 \cdots \sigma_j\sigma_{j-1}\sigma_j \cdots \sigma_{n-1} \overset{(1.6)}{=}$$
$$\sigma_1 \cdots \sigma_{j-1}\sigma_j\sigma_{j-1} \cdots \sigma_{n-1} \overset{(1.5)}{=} (\sigma_1\sigma_2 \cdots \sigma_{n-1})\sigma_{j-1}. \tag{1.7}$$

Nyní postupně indukcí ukážeme, že všechna slova ve znění lemmatu jsou ekvivalentní tomu definičnímu, tedy slovu

$$\Delta_n = (\sigma_1 \cdots \sigma_{n-1}) \cdots (\sigma_1\sigma_2)(\sigma_1) = (\sigma_1 \cdots \sigma_{n-1})\Delta_{n-1}. \tag{1.8}$$

Začněme slovem (1.1). Pro $n \in \{2,3\}$ platí ekvivalence slov triviálně, dále nechť $n > 3$. Využitím relace (1.5) přesuneme první člen každé závorky co nejblíže začátku slova a využijeme indukční předpoklad (IP), tedy

$$(\sigma_1)(\sigma_2\sigma_1)(\sigma_3\sigma_2\sigma_1) \cdots (\sigma_{n-1} \cdots \sigma_1)$$
$$\overset{(1.5)}{=} (\sigma_1\sigma_2\sigma_3 \cdots \sigma_{n-1})(\sigma_1)(\sigma_2\sigma_1) \cdots (\sigma_{n-2} \cdots \sigma_1)$$
$$\overset{(IP)}{=} (\sigma_1\sigma_2\sigma_3 \cdots \sigma_{n-1})\Delta_{n-1} \overset{(1.8)}{=} \Delta_n,$$

Pro slovo (1.2) máme pro $n = 2$ ekvivalenci triviálně. V indukčním kroku využijeme vztahu (1.7) a dostaneme

$$(\sigma_{n-1})(\sigma_{n-2}\sigma_{n-1}) \cdots (\sigma_1 \cdots \sigma_{n-1}) = (\sigma_1 \cdots \sigma_{n-1})(\sigma_{n-2})(\sigma_{n-3}\sigma_{n-2}) \cdots (\sigma_1 \cdots \sigma_{n-2}).$$

Použitím IP a (1.8) dostáváme, že i toto slovo popisuje Δ_n.

Nyní najednou dokážeme ekvivalenci slov (1.3),(1.4) a (1.8). Pro $n = 4$ a $i = 2$ upravíme slovo (1.4) pomocí relací:

$$\sigma_2\sigma_3\sigma_2\sigma_1\sigma_2\sigma_3 \overset{(1.6)}{=} \sigma_2\sigma_3\sigma_1\sigma_2\sigma_1\sigma_3 \overset{(1.5)}{=} \sigma_2\sigma_1\sigma_3\sigma_2\sigma_3\sigma_1 \overset{(1.6)}{=}$$
$$\sigma_2\sigma_1\sigma_2\sigma_3\sigma_2\sigma_1 \overset{(1.6)}{=} \sigma_1\sigma_2\sigma_1\sigma_3\sigma_2\sigma_1 \overset{(1.5)}{=} \sigma_1\sigma_2\sigma_3\sigma_1\sigma_2\sigma_1 \overset{(1.8)}{=} \Delta_n,$$

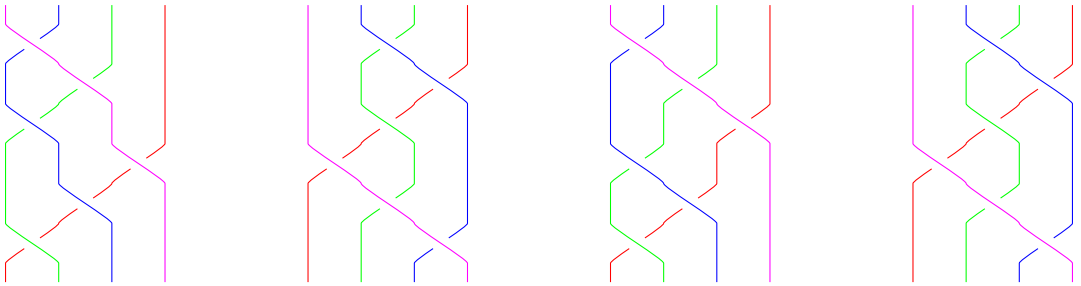
kde poslední slovo je zároveň i slovo (1.3). Pro $n > 4$ pomocí (1.7) ve slově (1.4) přesuneme člen $(\sigma_1 \cdots \sigma_{n-1})$ na začátek a dostaneme

$$\begin{aligned}&(\sigma_i)(\sigma_{i+1}\sigma_i)\cdots(\sigma_{n-1}\cdots\sigma_i)(\sigma_{i-1}\cdots\sigma_{n-1})\cdots(\sigma_1\cdots\sigma_{n-1})\\&\quad=(\sigma_1\cdots\sigma_{n-1})(\sigma_{i-1})(\sigma_i\sigma_{i-1})\cdots(\sigma_{n-2}\cdots\sigma_{i-1})(\sigma_{i-2}\cdots\sigma_{n-2})\cdots(\sigma_1\cdots\sigma_{n-2}).\end{aligned}$$

Pokud $i > 2$, tak použijeme IP pro $n-1$ a $i-1$ a dostaneme ekvivalenci všech tří slov; pro $i = 2$ využijeme slovo (1.2) $\Delta_{n-1} = (\sigma_1)(\sigma_2\sigma_1)\cdots(\sigma_{n-2}\cdots\sigma_1)$.

□

Alternativně si lze platnost lemmatu rozmyslet ryze geometricky. Náhled na tuto geometrickou interpretaci pro B_4 je na obrázku 1.9.



slovo (1.1) slovo (1.2) slovo (1.3) slovo (1.4)

Obrázek 1.9: Δ_4

Nyní se podíváme na několik vlastností fundamentálního copánku, které později využijeme při konstrukci normální formy.

Lemma 2. *Pro $n \in \mathbb{N}$ a $1 \leq i \leq n-1$ platí*

$$\Delta_n \sigma_i^m = \sigma_{n-i}^m \Delta_n, \tag{1.9}$$
$$\Delta_n^{-1} \sigma_i^m = \sigma_{n-i}^m \Delta_n^{-1}, \tag{1.10}$$

kde $m \in \{-1, 1\}$.

Důkaz.

Začneme důkazem pro pozitivní křížení tedy $m = 1$. Rovnost (1.9) nejdřív ukážeme pro případ $1 < i < n-1$ využitím slova (1.3) z lemmatu 1, tj.

$$\Delta_n = (\sigma_1 \cdots \sigma_{n-1}) \cdots (\sigma_1 \cdots \sigma_{i+1})(\sigma_1)(\sigma_2\sigma_1) \cdots (\sigma_i \cdots \sigma_1). \tag{1.11}$$

Stačí si rozmyslet následující dvě rovnosti, které získáme kombinací obou relací (případně snadnou indukcí). Pro $1 < i < n-1$ platí

$$(\sigma_1)(\sigma_2\sigma_1)\cdots(\sigma_i\cdots\sigma_1)\sigma_i = \sigma_1(\sigma_1)(\sigma_2\sigma_1)\cdots(\sigma_i\cdots\sigma_1), \tag{1.12}$$
$$(\sigma_1\cdots\sigma_{n-1})\cdots(\sigma_1\cdots\sigma_{i+1})\sigma_1 = \sigma_{n-i}(\sigma_1\cdots\sigma_{n-1})\cdots(\sigma_1\cdots\sigma_{i+1}). \tag{1.13}$$

Dosazením slova (1.11) a aplikací obou rovností výše dostaneme požadované $\Delta_n\sigma_i = \sigma_{n-i}\Delta_n$. Krajní případy, tedy $i \in \{1, n-1\}$, dostaneme přímo z rovností (s tím, že (1.12) platí i pro $i = n-1$ a (1.13) i pro $i = 1$), kde pro důkaz $i = 1$ využijeme slovo (1.2) $\Delta_n = (\sigma_{n-1})(\sigma_{n-2}\sigma_{n-1})\cdots(\sigma_1\cdots\sigma_{n-1})$ a pro $i = n-1$ použijeme slovo (1.1) $\Delta_n = (\sigma_1)(\sigma_2\sigma_1)\cdots(\sigma_{n-1}\cdots\sigma_1)$.

Rovnost (1.10) získáme položením $i = n - j$ a přidáním Δ_n^{-1} zleva a zprava na obě strany rovnosti 1.9:

$$\begin{aligned}
\Delta_n\sigma_j &= \sigma_{n-j}\Delta_n \\
\Delta_n^{-1}\Delta_n\sigma_j\Delta_n^{-1} &= \Delta_n^{-1}\sigma_{n-j}\Delta_n\Delta_n^{-1} \\
\sigma_j\Delta_n^{-1} &= \Delta_n^{-1}\sigma_{n-j} \\
\sigma_{n-i}\Delta_n^{-1} &= \Delta_n^{-1}\sigma_i.
\end{aligned}$$

Platnost pro negativní křížení, tj. pro $m = -1$, dostáváme invertováním obou stran rovností (1.9),(1.10) s $m = 1$.

□

Značení. Pro copánek $b = \sigma_{i_1}^{m_1}\sigma_{i_2}^{m_2}\cdots\sigma_{i_k}^{m_k}$ značíme $\pi_n(b) = \sigma_{n-i_1}^{m_1}\sigma_{n-i_2}^{m_2}\cdots\sigma_{n-i_k}^{m_k}$.

Důsledek. Pro $b \in B_n$ platí $\Delta_n b = \pi_n(b)\Delta_n$. Proto $\Delta_n^2 b = b\Delta_n^2$, tedy Δ_n^2 komutuje s libovolným copánkem.

Pozorování. Platí $\pi_n(\Delta_n) = \Delta_n$

Intuitivně si $\pi_n(b)$ můžeme představit jako copánek, který vidíme při pohledu na b z opačné strany. Příklad můžeme vidět na obrázku 1.10

Obrázek 1.10: copánky b a $\pi_n(b)$

Nyní si ukážeme, jak lze pomocí fundamentálního copánku zapsat libovolné negativní křížení.

Tvrzení 3. *Nechť $n \in \mathbb{N}$. Pak $\forall i \in \{1, \ldots, n-1\}$ $\exists B \in B_n^+$: $\sigma_i^{-1} = \Delta_n^{-1}B$.*

Důkaz. Pro $i = 1$ využijeme pro popsání Δ_n^{-1} definují slovo a tvar inverzu, který jsme ukázali po zavedení copánkové grupy. Máme

$$\Delta_n^{-1} = (\sigma_1^{-1})(\sigma_2^{-1}\sigma_1^{-1})\cdots(\sigma_{n-1}^{-1}\cdots\sigma_1^{-1})$$

a hledaný pozitivní copánek je pak zjevně právě

$$B = (\sigma_1\cdots\sigma_{n-1})\cdots(\sigma_1\sigma_2).$$

Pro $1 < i < n-1$ využijeme slovo (1.4), tj.

$$\Delta_n^{-1} = (\sigma_{n-1}^{-1}\cdots\sigma_1^{-1})\cdots(\sigma_{n-1}^{-1}\cdots\sigma_{i-1}^{-1})(\sigma_i^{-1}\cdots\sigma_{n-1}^{-1})\cdots(\sigma_i^{-1}\sigma_{i+1}^{-1})(\sigma_i^{-1})$$

Snadno najdeme $B' \in B_n^+$, že $\sigma_i^{-1} = B'\Delta_n^{-1}$ a použitím lemmatu 2 dostaneme hledané $B = \pi_n(B')$.

Pro $i = n-1$ postupujeme analogicky využitím slova (1.2), tj.

$$\Delta_n = (\sigma_{n-1})(\sigma_{n-2}\cdots\sigma_{n-1})\cdots(\sigma_1\cdots\sigma_{n-1})$$

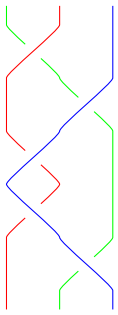
□

Příklad negativního křížení zapsaného pomocí inverzu fundamentálního copánku máme na obrázku 1.11.

Předchozí tvrzení nám dává návod, jak libovolný copánek $b \in B_n$ „zbavit“ negativních křížení. Všechna negativní křížení přepíšeme pomocí fundamentálních copánků, které pak použitím lemmatu 2 přesuneme na začátek slova. Tedy dostaneme

$$b = \Delta_n^r B,$$

pro nějaký pozitivní copánek $B \in B_n^+$ a $r < 0$. Zbývá jen zavést formát zápisu pro pozitivní copánky, k čemuž budeme potřebovat zavést takzvané *permutační copánky*.



Obrázek 1.11: $\sigma_1^{-1} = \Delta_n^{-1}\sigma_1\sigma_2$

1.1.2 Permutační copánky

V této sekci se vrátíme ke geometrickému pohledu na copánky. Provázky v copánku mohou končit v libovolných různých bodech, a jejich proházení tak lze popsat pomocí permutace. Navíc je zřejmé, že v symetrické grupě S_n platí pro transpozici $\tau_i = (i\ i+1)$ následující

$$\begin{aligned} \tau_i\tau_j &= \tau_j\tau_i, && |i-j| > 1 \\ \tau_i\tau_{i+1}\tau_i &= \tau_{i+1}\tau_i\tau_{i+1}, \\ \tau_i\tau_i^{-1} &= id. \end{aligned}$$

Máme tedy surjektivní homomorfismus $\phi\colon B_n \twoheadrightarrow S_n$ takový, že $\forall i \in \{1,\ldots,n-1\}$: $\phi(\sigma_i) = \phi(\sigma_i^{-1}) = (i\ i+1)$. Pokud $\phi(b) = \pi$, pak říkáme, že copánek b *indukuje*

permutaci π. Jádrem ϕ jsou takzvané *ryzí copánky*, s nimiž budeme krátce pracovat v druhé kapitole.

Bude užitečné zavést kanonický inverz k ϕ, který bude určovat právě množinu permutačních copánků. Existuje několik ekvivalentních definic, pro nás však bude nejvhodnější geometrická definice převzatá z (Elrifai a Morton, 1994).

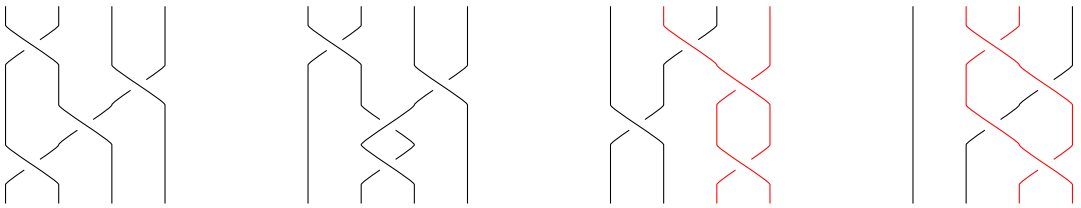
Definice 4. *Pozitivní copánek $b \in B_n^+$ nazveme* permutačním copánkem *pokud je možné jej v rovině nakreslit jako copánek, v němž se každý pár provázků kříží nanejvýš jednou.*

Poznámka. Pro permutační copánek nakreslení, kde se každý pár provázků kříží nanejvýš jednou, získáme z pozitivního slova, které jej popisuje.

Poznámka. Nadále budeme pro pozitivní copánky uvažovat pouze zápisy pozitivním slovem; řekneme tedy, že v copánku $b \in B_n^+$ se dva provázky *kříží*, pokud se kříží při nakreslení nějakého pozitivního slova popisujícího b — pro všechna pozitivní slova popisující b je počet křížení dvou daných provázků stejný. Například na obrázku 1.12 se v druhém copánku zleva první a čtvrtý provázek nekříží, jelikož se nekříží při zápisu copánku pozitivním slovem.

Značení. Množinu permutačních copánků budeme značit $\tilde{S}_n$.

Na obrázku 1.12 vidíme příklady copánků, kde provázky jsou červené, pokud se kříží dvakrát (tedy copánky s červenými provázky nejsou permutační).



$\sigma_1\sigma_3\sigma_2\sigma_1$ $\quad$ $\sigma_2\sigma_2^{-1}\sigma_3\sigma_1 = \sigma_3\sigma_1$ $\quad$ $\sigma_2\sigma_3\sigma_1\sigma_3$ $\quad$ $\sigma_2\sigma_3\sigma_2\sigma_3$

Obrázek 1.12: Vlevo permutační copánky; vpravo copánky, které nejsou permutační

Nyní si rozmyslíme, že permutační copánky jsou isomorfní symetrické grupě. V důkazu budeme postupovat analogicky jako (Elrifai a Morton, 1994).

Tvrzení 4. *Zobrazení $\phi \restriction \tilde{S}_n \colon \tilde{S}_n \to S_n$ je bijekce.*

Důkaz. Nejdřív ukážeme, že zobrazení je prosté, čili nechť $A_1, A_2 \in \tilde{S}_n$ a oba indukují permutaci π, pak chceme $A_1 = A_2$. Označme provázky čísly $\{1, \ldots, n\}$

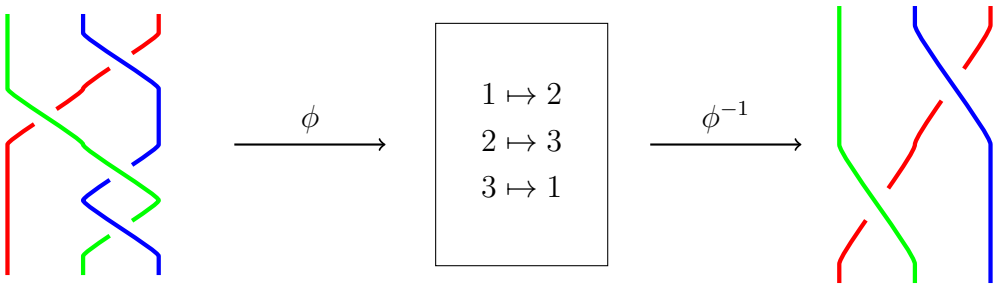
podle jejich začínající pozice. V permutačním copánku se pro $i < j$ i-tý provázek kříží s j-tým nanejvýš jednou, a to pozitivně — tedy i-tý provázek je nad j-tým. Proto první provázek bude nad všemi ostatními, druhý nad všemi krom prvního a tak dále. Permutační copánky tak můžeme v $\mathbb{R}^3$ zakreslit s každým provázkem ve vlastní rovině. Jelikož A_1, A_2 indukují stejnou permutaci, tak v každé rovině bude daný provázek pro oba copánky končit ve stejném bodě, a tudíž jsou A_1 a A_2 stejné copánky.

Dále si potřebujeme rozmyslet, že ke každé permutaci π existuje permutační copánek, který ji indukuje. Uspořádejme n bodů na horní a dolní hraně obdélníku a nakresleme čáry spojující i-tý bod nahoře s $\pi(i)$-tým bodem dole tak, že se každé dvě kříží nanejvýš jednou a žádné tři se nekříží v jednom bodě. Shora pak postupně přepíšeme každé křížení na σ_i a získáme permutační copánek indukující permutaci π.

□

Značení. Permutační copánek indukující permutaci π budeme značit $\phi^{-1}(\pi)$.

Zobrazení ϕ a jeho kanonický inverz ilustruje obrázek 1.13.



Obrázek 1.13: Ukázka homomorfismu ϕ a jeho kanonického inverzu

V následujících dvou lemmatech ukážeme, jak s permutačními copánky souvisí fundamentální copánek.

Lemma 5. *Pro $n \in N$ je fundamentální copánek Δ_n permutační copánek a indukuje permutaci $\pi_n \in S_n$: $\pi_n(i) = n - i + 1$.*

Důkaz. Již jsme si rozmysleli, že při nakreslení Δ_n se všechny provázky kříží právě jednou, tudíž jde o permutační copánek. Podobu indukované permutace dokážeme indukcí pomocí vztahu $\Delta_n = (\sigma_1 \cdots \sigma_{n-1})\Delta_{n-1}$.

Pro $n = 2$ copánek $\Delta_2 = \sigma_1$ indukuje $\pi_2 = (1\ 2)$, dále nechť $n > 2$. Copánek $(\sigma_1 \cdots \sigma_{n-1})$ indukuje permutaci $\rho \in S_n$: $(\rho(1) = n) \wedge (\forall i \neq 1\colon \rho(i) = i - 1)$ a dále z indukčního předpokladu máme, že copánek Δ_{n-1} indukuje permutaci $\pi_{n-1} \in S_n$: $(\pi_{n-1}(n) = n) \wedge (\forall i \leq n - 1\colon \pi_{n-1}(i) = n - i)$. Složení copánků tak

indukuje permutaci $\pi_{n-1} \circ \rho = \pi_n$.

□

Lemma 6. *Každý permutační copánek můžeme zprava i zleva doplnit na fundamentální copánek permutačním copánkem, tj.* $\forall B \in \tilde{S}_n \colon (\exists C_1 \in \tilde{S}_n \colon BC_1 = \Delta_n) \wedge (\exists C_2 \in \tilde{S}_n \colon C_2 B = \Delta_n)$.

Důkaz. Nejdřív si uvědomíme, že pokud $A, B \in B_n^+$ takové, že $AB = \Delta_n$, pak se v copáncích A, B každý pár provázků kříží nanejvýš jednou, jelikož v Δ_n se každé dva provázky kříží právě jednou. Tudíž $A, B \in \tilde{S}_n$.

Nyní si dokážeme doplnění zprava. Pro permutaci $\pi = \phi(B) \in S_n$ snadno najdeme $\rho \in S_n$, že $\rho \circ \pi = \pi_n$, kde $\pi_n = \phi(\Delta_n)$, tj. $\forall i \colon \pi_n(i) = n - i + 1$. Položme $C_1 = \phi^{-1}(\rho)$. Pak copánek BC_1 jistě indukuje permutaci π_n, navíc se v něm každé dva provázky kříží nanejvýš dvakrát, protože B i C_1 jsou permutační copánky. Navíc $\forall i < j \colon \pi_n(i) > \pi_n(j)$, a tedy pokud i-tý provázek začíná před j-tým, tak musí končit za ním. Proto se v copánku, který na provázcích indukuje π_n, musí každé dva provázky křížit lichý počet krát, již jsme si ale rozmysleli, že se kříží nanejvýš dvakrát. Dohromady se tak v BC_1 se každé dva provázky kříží právě jednou, a tudíž $BC_1 = \Delta_n$.

Doplnění zleva lze dokázat analogicky.

□

1.1.3 Normální sekvence

V této sekci zavedeme jednotný zápis pro pozitivní copánky, k čemuž budeme potřebovat další definici.

Definice 5. *Pro* $B \in B_n^+$ *definujeme* začínající množinu $S(B) \subseteq \{1, \ldots, n-1\}$*:*

$$S(B) = \{i \mid \exists B' \in B_n^+ \colon B = \sigma_i B'\},$$

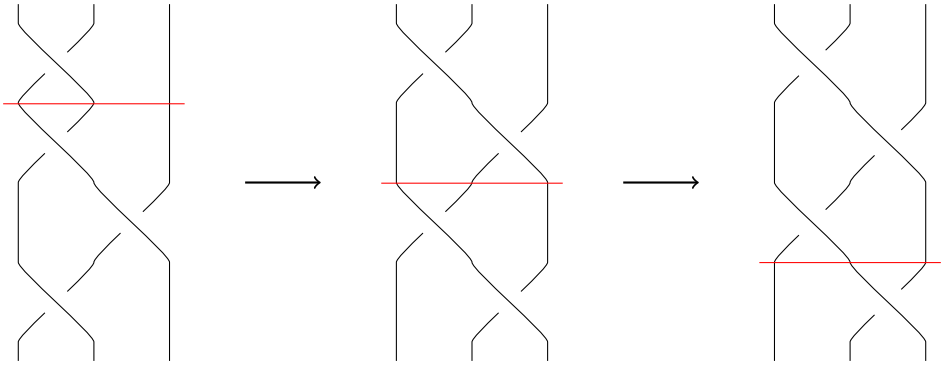
a koncovou množinu $F(B) \subseteq \{1, \ldots, n-1\}$*:*

$$F(B) = \{i \mid \exists B' \in B_n^+ B = B' \sigma_i\}.$$

Definice 6. $p, n \in \mathbb{N}$. *Složení permutačních copánků* $A_1 \cdots A_p$ *nazveme* normální sekvencí*, pokud pro každé* $i \in \{1, \ldots, p-1\}$ *platí* $S(A_{i+1}) \subseteq F(A_i)$.

Poznámka. Podmínka $S(A_{i+1}) \subseteq F(A_i)$ v předchozí definici odpovídá tomu, že A_i je maximální permutační copánek, kterým začíná copánek $A_i A_{i+1}$. To znamená, že $\forall i \in S(A_{i+1})$ copánek $A_i \sigma_i$ již není permutační.

Na obrázku 1.14 vidíme příklad, jak různé dělení pozitivního copánku na dva permutační ovlivňuje jejich začínající a koncové množiny.



Obrázek 1.14: $\sigma_1\sigma_1\sigma_2\sigma_1 = \sigma_1\sigma_2\sigma_1\sigma_2$
Červená čára ukazuje kde končí levý permutační copánek

Vlevo máme copánek AB, kde $F(A) = \{1\},\ S(B) = \{1,2\}$, uprostřed pak $F(A') = \{2\},\ S(B') = \{1\}$, a vpravo $F(A'') = \{1,2\},\ S(B'') = \{2\}$, a tedy $A''B''$ je normální sekvence.

Nyní již konečně máme prostředky pro definici normální formy.

1.1.4 Normální forma

Definice 7. *Řekneme, že copánek $b \in B_n$ je v* normální formě*, pokud je daný slovem:*

$$b = \Delta_n^u A_1 A_2 \cdots A_p,$$

kde $u \in \mathbb{Z}, p \in \mathbb{N}_0$ a $A_1 A_2 \cdots A_p$ je normální sekvence, pro kterou navíc platí $\forall i \in \{1,\ldots,p\}\colon A_i \neq \Delta_n$. Číslu u pak říkáme mocnina *copánku.*

Pro uložení copánku v normální formě nám stačí celé číslo u a posloupnost permutací. Navíc krom inverzu fundamentálního copánku zápis neobsahuje negativní křížení, což může usnadnit důkazy mnoha vět, jelikož stačí pracovat s pozitivními copánky.

Poznámka. V některých pozdějších důkazech budeme normální formu psát jako $b = \Delta_n^u B$ pro $B \in B_n^+$, jelikož nebude nutné pracovat s konkrétní podobou B.

Nejzásadnější vlastností normální formy je její jednoznačnost. Bohužel je důkaz jednoznačnosti značně technický, a proto si z (Elrifai a Morton, 1994) pouze vypůjčíme následující větu.

Věta 7. *Každý copánek lze převést do normální formy a tento zápis je jednoznačný.*

Pozorování. Jednoznačnost normální formy nám umožňuje ověřit, zda dvě různá slova popisují stejný copánek. Stačí obě převést do normální formy a následně je porovnat.

Ukážeme si algoritmus, jak libovolný copánek převést do jeho normální formy. Aby však dával smysl, potřebujeme dokázat další lemma týkající se permutačních copánků.

Lemma 8. *Pro permutační copánek $B = \phi^{-1}(\pi)$ jsou následující tvrzení ekvivalentní*

(1) $i \in S(B)$,

(2) provázky i *a* $i+1$ *se kříží v* B,

(3) $\pi(i+1) < \pi(i)$.

Důkaz. (2) $\iff$ (3): zřejmé, jelikož i-tý a $(i+1)$-ní provázky se mohou b B křížit nanejvýš jednou.

(1) $\implies$ (2): Můžeme psát $B = \sigma_i B'$ pro nějaké $B' \in B_n^+$. Provázky i a $i+1$ se kříží v σ_i, tudíž se jistě kříží i v B.

(3) $\implies$ (1): Jistě existuje $\rho \in S_n$ taková, že $\pi = \rho \circ \tau_i$, kde $\tau_i = (i\ i+1)$. Pak platí $\rho(i+1) > \rho(i)$, jelikož $\pi(i+1) < \pi(i)$ a $\tau_i(i+1) < \tau_i(i)$. Uvažujme permutační copánek $B' = \phi^{-1}(\rho)$, pak z předchozího a z ekvivalence (2) $\iff$ (3) máme, že v B' se provázky začínající na i-té a $(i+1)$-ní pozici nekříží. Copánek $\sigma_i B'$ pak indukuje permutaci $\rho \circ \tau_i = \pi$ a provázky i a $i+1$ se v něm kříží právě jednou, tudíž je to permutační copánek. To ale znamená, že $\sigma_i B' = B$, jelikož permutační copánek indukující danou permutaci je dle tvrzení 4 ($\tilde{S_n} \simeq S_n$) určen jednoznačně.

□

Nyní ještě zformulujeme analogické tvrzení pro koncovou množinu, které lze dokázat analogickým způsobem.

Důsledek. Uvažujme permutační copánek $B = \phi^{-1}(\pi)$ a nechť k-tý provázek končí v B na i-té pozici a l-tý provázek na $(i+1)$-ní pozici, tj. $\pi(k) = i, \pi(l) = i+1$. Pak jsou následující tvrzení ekvivalentní

(1) $i \in F(B)$,

(2) k-tý a l-tý provázek se kříží v B,

(3) $k > l$.

Důsledek. Pokud $A \in \tilde{S_n}$ a $i \notin S(A)$, pak platí $\sigma_i A \in \tilde{S_n}$. Analogicky pokud $i \notin F(A)$, pak $A\sigma_i \in \tilde{S_n}$.

Algoritmus 1 Garsideův–Thurstonův algoritmus

Vstup: $b \in B_n : b = \sigma_{i_1}^{m_1}\sigma_{i_2}^{m_2}\cdots\sigma_{i_{|b|}}^{m_{|b|}}$, kde $m_i \in \{1, -1\}$
Výstup: Normální forma $b = \Delta_n^u A_1 A_2 \cdots A_p$ z definice 7

(1) Všechna negativní křížení přepiš pomocí tvrzení 3 na $\sigma_i^{-1} = \Delta_n^{-1} A$, pro nějaký pozitivní copánek $A \in B_n^+$.

(2) Pomocí lemmatu 2 přesuň všechny fundamentální copánky vzniklé v kroku (1) na začátek slova. Dostaneme $b = \Delta_n^{u'} d$, pro nějaký pozitivní copánek $d \in B_n^+$.

(3) Rozděl d na permutační copánky: $d = A_1 A_2 \cdots A_{p'}$.

(4) **while** d není v normální formě **do**

  (a) Pro každé A_i urči $S(A_i)$ a $F(A_i)$.

  (b) Najdi v d první dvojici $A_i A_{i+1}$ splňující $S(A_{i+1}) \not\subseteq F_(A_i)$ a vezmi libovolné $j \in S(A_{i+1}) \backslash F(A_i)$.

  (c) Polož $B_i = A_i \sigma_j$ a $B_{i+1} = \sigma_j^{-1} A_{i+1}$.

  (d) v d nahraď $A_i A_{i+1} \leftarrow B_i B_{i+1}$.

  (e) Pomocí lemmatu 2 přesuň všechna $A_i = \Delta_n$ na začátek slova a vypusť všechna $A_j = \varepsilon$.

(5) **return** $\Delta_n^u A_1 A_2 \cdots A_p$

Poznámka. Dle důsledku před algoritmem jsou copánky B_i v kroku (4c) permutační. B_{i+1} jsou také permutační, jelikož můžeme psát $A_{i+1} = \sigma_i A'$, kde $A' \in \tilde{S}_n$, a tudíž $B_{i+1} = A'$. Dohromady jsou tedy A_i vždy permutační copánky, navíc díky kroku (4e) $A_i \neq \Delta_n$, což vyžadujeme v definici normální formy 7.

V kroku (4d) se vždy přiblížíme k normální formě. Navíc existuje jen konečně mnoho způsobů jak pozitivní copánek d zapsat jako složení permutačních copánků, protože existuje jen konečně pozitivních slov, kterými lze d popsat. A tedy počet kroků algoritmu je vždy konečný.

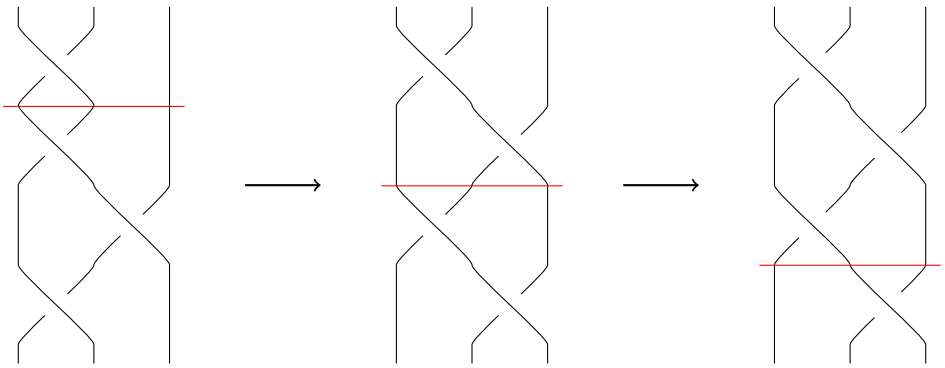
Za pozornost stojí především krok (4a) — určení množin $S(A)$ a $F(A)$. Je zřejmé, že pro libovolný copánek jde o značně netriviální úlohu, v algoritmu však tyto množiny určujeme pouze pro permutační copánky, kterých je konečný počet (pro B_n konkrétně $n!$). (Elrifai a Morton, 1994) tak navrhují uložit $S(A)$ a $F(A)$ pro každý permutační copánek $A \in \tilde{S}_n$. Při vhodné implementaci musíme uložit $2(n-1)n!$ celých čísel, a prostorová složitost je tedy $O(n!)$. Časová složitost kroku (4a) je pak však konstantní, protože stačí určit $S(A_i)$ a $F(A_i)$ dvou permutačních

copánků, které se v předchozím kole cyklu změnily. Alternativně můžeme využít lemma 8 o podobě $S(A)$ a $F(A)$ pro $A \in \tilde{S}_n$, čili ověřovat, které provázky se v daném copánku kříží. Tento přístup je na druhou stranu časově náročnější, jde tedy o klasický time-memory trade-off.

Podobně důležitá je i správná implementace kroku (3). Aby algoritmus vrátil správný výstup v polynomiálním čase, můžeme slovo rozdělit na permutační copánky libovolně — např. každý generátor odpovídá jednomu permutačnímu copánku, tj. pro $d = \sigma_{j_1}\sigma_{j_2}\cdots\sigma_{j_{|d|}}$ volíme $A_i = \sigma_{j_i}$. Snadno si však rozmyslíme, že kvůli kroku (1) může pro copánek d s pouze pozitivními kříženími nastat až $|d| = |b| \cdot \left(\frac{n(n-1)}{2} - 1\right)$. Výrazně vhodnější je tak volit nejdelší možné permutační copánky, tedy zvolit $A_1 = \sigma_{j_1}\sigma_{j_2}\cdots\sigma_{j_l}$ tak, že $A_1\sigma_{j_{l+1}}$ už není permutační copánek. Další A_i pak volíme analogicky. Toto rozdělení proběhne v lineárním čase a počet permutačních copánků bude $p' \leq |b|$, jelikož v nahrazení $\sigma_i^{-1} = \Delta_n^{-1}A$ z tvrzení 3 je $A \in B_n^+$ permutační copánek (lze jej doplnit na Δ_n, tudíž se v něm každý pár provázků kříží nanejvýš jednou).

Dle (Birman a kol., 1998) pracuje algoritmus při správné implementaci v čase $\mathcal{O}(|b|^2\, n \log n)$, kde $|b|$ je délka slova a n je počet provázků.

Průběh algoritmu si ukážeme na obrázku 1.15 pro copánek $b = \sigma_1\sigma_1\sigma_2\sigma_1$.



Obrázek 1.15: $b = \sigma_1\sigma_1\sigma_2\sigma_1 = \sigma_1\sigma_2\sigma_1\sigma_2$
Červená čára ukazuje, kde končí levý permutační copánek

V krocích (1) a (2) se slovo $b = \sigma_1\sigma_1\sigma_2\sigma_1$ nijak nezmění, jelikož jsou všechna křížení pozitivní. Následně v kroku (3) b rozdělíme na permutační copánky postupem popsaným výše na $b = A_1A_2$, kde $A_1 = \sigma_1, A_2 = \sigma_1\sigma_2\sigma_1$ (viz obrázek vlevo). Snadno určíme, že $F(A_1) = \{1\}$, $S(A_2) = \{1, 2\}$, tedy A_1A_2 není normální sekvence, a tudíž b není v normální formě. V kroku (4b) tak zvolíme $2 \in S(A_2)\backslash F(A_1)$, pak v (4c) položíme $B_1 = \sigma_1\sigma_2$, $B_2 = \sigma_2^{-1}\sigma_1\sigma_2\sigma_1 = \sigma_1\sigma_2$ a nakonec v (4d) nahradíme $A_1A_2 \leftarrow B_1B_2$. Máme tedy $b = A_1A_2$ pro $A_1 = \sigma_1\sigma_2, A_2 = \sigma_1\sigma_2$, jak vidíme na obrázku uprostřed. Copánek stále není v normální formě, jelikož

$S(A_2) = \{1\} \not\subseteq \{2\} = F(A_1)$, ale zopakováním kroku (4) dostaneme copánek na obrázku vpravo, který již je v normální formě.

2. Word a conjugacy problem

V této kapitole si popíšeme dva základní problémy týkající se copánkových grup, které našly využití v kryptografii. Ukážeme si původní řešení obou problémů a pouze stručně nastíníme myšlenky komplexnějších řešení, které se dají efektivně využít v praxi. Několik vět, které v této kapitole vyslovíme, tak bude ponecháno bez důkazu, pouze se odkážeme na příslušné zdroje.

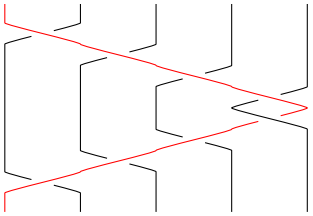
V následujícím textu budeme pracovat v grupě B_n pro $n \in \mathbb{N}$.

2.1 Word problem

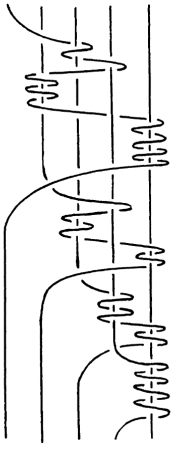
Nejzásadnější úlohou v copánkových grupách je rozpoznat, zda dvě různá slova popisují ten samý copánek. Tato úloha se nazývá *word problem*.

Jak si brzy ukážeme, tento problém není dost obtížný, aby na něm bylo možné vybudovat kryptografická schémata, je však zásadní jej umět řešit co nejefektivněji. Ve schématech totiž budou účastníci často pracovat se stejným copánkem, který však bude popsaný různými slovy — abychom skutečně mohli říct, že jde o ten samý copánek, musíme umět řešit *word problem*.

Existuje několik řešení, z nichž většina stojí na stejném principu využívajícím nějakou jednoznačnou formu zápisu copánku — *word problem* pak vyřešíme převodem obou copánků do této jednoznačné formy a jejich následným porovnáním. První tři řešení, které v této práci nabídneme, se tak liší pouze ve zvolené formě.



Obrázek 2.1: $A_{1,5}$



Obrázek 2.2: Artinova jednoznačná forma

2.1.1 Artinovo řešení

První řešení úlohy nabídl hned (Artin, 1947), který k získání jednoznačné formy pracoval s ryzími copánky, tj. s jádrem zobrazení $\phi\colon B_n \twoheadrightarrow S_n$ (copánky, které nemění pořadí provázků). Využil generátory A_{ij} — i-tý provázek kříží j-tý provázek — viz obrázek 2.1. Pro ryzí copánek R pak definoval jednoznačný zápis:

$$R = U_1 U_2 \cdots U_{n-1},$$

kde U_i je složení mocnin generátorů A_{ij} takových, že $i < j$. Postupně tak popisujeme všechna křížení prvního copánku, pak druhého a tak dále. Geometrická

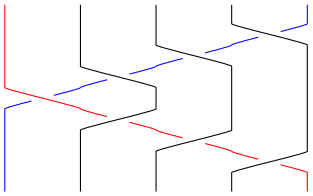
interpretace této formy je na obrázku 2.2 převzatého z (Artin, 1947). Copánky, které nejsou ryzí, lze vyjádřit jako složení copánku v této formě a vhodného permutačního copánku. Artin dokázal jednoznačnost formy, ale žádný algoritmus pro převod ve článku neposkytl.

2.1.2 Řešení pomocí fundamentálního copánku

Dokázané řešení v polynomiálním čase nabídl až Thurston navazující svou prací na Garsideovy výsledky. Garsideův–Thurstonův algoritmus 1 řeší *word problem* v čase $\mathcal{O}(|b|^2\, n \log n)$, kde $|b|$ je délka slova v artinovském zápisu. Tento algoritmus společně s teorií za ním jsme představili v předchozí kapitole.

Ještě efektivnější řešení nabízí (Birman a kol., 1998), kteří pracovali s takzvanými *band generators* — pro $i > j$ máme $\alpha_{ij} = (\sigma_{i-1} \cdots \sigma_{j+1})\sigma_j(\sigma_{j+1}^{-1} \cdots \sigma_{i-1}^{-1})$.

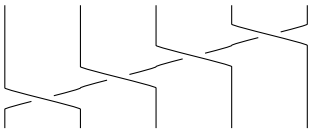
Z geometrického pohledu jde o prohození i-tého provázku s j-tým, což můžeme vidět na obrázku 2.3 pro generátor α_{51}. Tyto prvky generují B_n pomocí podobně jednoduchých relací jako při artinovské prezentaci. (Birman a kol., 1998) v článku pracovali s alternativním fundamentálním copánkem (viz obrázek 2.4):



Obrázek 2.3: α_{51}

$$\delta_n = \alpha_{n\,n-1} \cdots \alpha_{3\,2}\alpha_{2\,1} = \sigma_{n-1} \cdots \sigma_2\sigma_1.$$

V následujícím tvrzení si vykreslíme paralelu mezi Δ_n a δ_n, tedy nastíníme, že mohou mít určité podobné vlastnosti.



Obrázek 2.4: δ_5

Tvrzení 9. *Platí* $\delta_n^n = \Delta_n^2$.

Důkaz. Nejdřív připomeňme, že pro libovolný $b \in B_n$: $b = \sigma_{i_1}^{m_1}\sigma_{i_2}^{m_2} \cdots \sigma_{i_k}^{m_k}$ značíme $\pi_n(b) = \sigma_{n-i_1}^{m_1}\sigma_{n-i_2}^{m_2} \cdots \sigma_{n-i_k}^{m_k}$. Důsledek lemmatu 2 říká $b\Delta_n = \Delta_n\pi_n(b)$, z čehož plyne $\Delta_n = \pi_n(\Delta_n)$. Využitím definičního slova $\Delta_n = (\sigma_1 \cdots \sigma_{n-1}) \cdots (\sigma_1\sigma_2)(\sigma_1)$ pak dostáváme $\pi_n(\Delta_n) = (\sigma_{n-1} \cdots \sigma_1) \cdots (\sigma_{n-1}\sigma_{n-2})(\sigma_{n-1})$. Navíc z lemmatu 1 máme i popisující slovo $\Delta_n = (\sigma_1)(\sigma_2\sigma_1) \cdots (\sigma_{n-1} \cdots \sigma_1)$. Složením těchto dvou slov a použitím relace $\forall|i-j| > 1$: $\sigma_i\sigma_j = \sigma_j\sigma_i$ můžeme ukázat

$$\begin{aligned}
\Delta_n^2 &= \pi_n(\Delta_n)\Delta_n = (\sigma_{n-1} \cdots \sigma_1) \cdots (\sigma_{n-1}\sigma_{n-2})(\sigma_{n-1})(\sigma_1)(\sigma_2\sigma_1) \cdots (\sigma_{n-1} \cdots \sigma_1) \\
&= (\sigma_{n-1} \cdots \sigma_1)(\sigma_{n-1} \cdots \sigma_2)(\sigma_1)(\sigma_{n-1} \cdots \sigma_3)(\sigma_2\sigma_1) \cdots (\sigma_{n-1} \cdots \sigma_1) \\
&= (\sigma_{n-1} \cdots \sigma_1)(\sigma_{n-1} \cdots \sigma_1) \cdots (\sigma_{n-1} \cdots \sigma_1) = \delta_n^n.
\end{aligned}$$

□

(Birman a kol., 1998) ukázali, že libovolný copánek $b \in B_n$ pak můžeme vyjádřit jednoznačným slovem:

$$b = \delta_n^r A_1 \dots A_p,$$

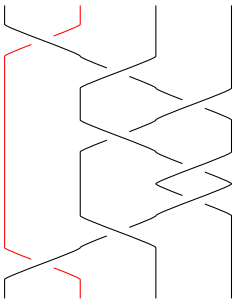
kde copánky A_i náleží do speciální podmnožiny permutačních copánků, jejíž konkrétní podoba je mimo záběr této práce.

Autoři článku nabídli algoritmus pracující v čase $\mathcal{O}(|W|^2\, n)$, tedy vylepšili Garsidův–Thurstonův algoritmus o faktor $\log n$. Navíc krom asymptotického zlepšení jejich algoritmus dosáhl i výrazně kratší faktické doby výpočtu. Důvodem, je že α_{ij} nahrazuje slovo zapsané pomocí $2(i-j)-1$ artinovských generátorů σ_i, a tudíž délka slova může být až n-krát menší než v artinovském zápisu.

2.1.3 Handle reduction

Připomeňme si složitost Garsideova–Thurstonova algoritmu 1, který převáděl copánek do jeho normální formy. Zatímco časová složitost je polynomiální, prostorová je při určité implementaci vzhledem k počtu provázků silně exponenciální, dokonce $\mathcal{O}(n!)$. Pro větší n se tak algoritmus stává takřka nepoužitelným. Algoritmus využívající *band generators* se podobnému problému vyhýbá, tudíž je i z tohoto pohledu výhodnější alternativou. Existuje však i metoda řešení *word problem*, která vůbec nezávisí na počtu provázků, čili jde zobecnit pro grupu B_∞.

Metoda, kterou popsal (Dehornoy, 1997), zobecňuje nejtriviálnější úpravu copánku, a to nahrazení $\sigma_i\sigma_i^{-1}$ prázdným slovem. Dehornoy tuto úpravu zobecnil pro takzvaný *úchop* (viz obrázek 2.5).



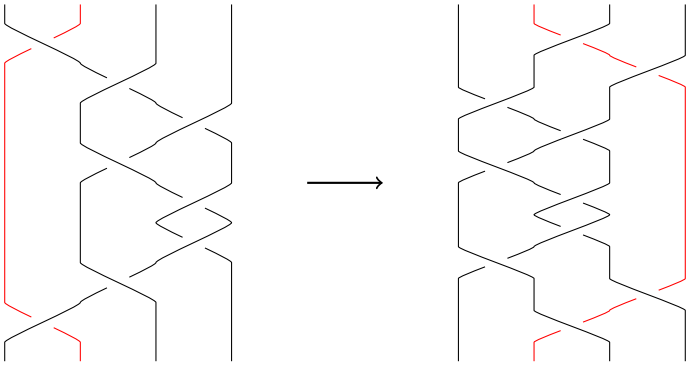
Obrázek 2.5: σ_1-úchop

Definice 8. *Slovo $\sigma_i^e v \sigma_i^{-e}$, kde $e \in \{-1, 1\}$ nazveme σ_i−úchop, pokud slovo v obsahuje pouze generátory $\sigma_j^{\pm 1}$ pro $j > i$.*

Řekneme, že copánek $b \in B_n$ je v plně redukované formě*, pokud slovo, které jej popisuje, neobsahuje žádný úchop.*

Tvrzení 10. *Neprázdný copánek v plně redukované formě nemůže být ekvivalentní prázdném slovu ε.*

Pomocí tohoto tvrzení, které jsme si vypůjčili z (Dehornoy, 1994), již můžeme řešit *word problem*. Platí totiž, že dvě slova b, b' odpovídají stejnému copánku, právě když $b^{-1}b' = \varepsilon$. Stačí tedy ověřit, zda plně redukovaná forma tohoto copánku odpovídá prázdnému slovu. Algoritmus pro převod do plně redukované formy popsaný v (Dehornoy, 1997) se nazývá *handle reduction* a jeho podstatou je eliminace úchopů. Příklad přirozeného způsobu řešení je na obrázku 2.6.



Obrázek 2.6: Handle reduction

Tato úprava však zjevně může prodloužit délku slova až o $2n$ generátorů. Navíc lze zkonstruovat slovo, v němž opakovanou aplikací úpravy vždy vznikne nový úchop. Algoritmus který poskytl (Dehornoy, 1997), tak využívá komplexnější postupy. Pro dokázání konečnosti využívá Dehornoy husté linearní uspořádání na copánkových grupách, které zavedl dříve ve svém článku (Dehornoy, 1994). Tento algoritmus pak pracuje v polynomiálním čase vzhledem k délce slova.

2.1.4 Shortest word problem a membership problem

Zdánlivě rozumným řešením *word problem* je převedení copánku do jeho nejkratšího možného zápisu. Tato úloha se nazývá *shortest word problem* a je ve skutečnosti výrazně obtížnější než běžný *word problem*. (Paterson a Razborov, 1991) dokonce ukázali, že jde o NP-těžký problém, tedy není jisté, jestli je vůbec možné v polynomiálním čase určit, zda je daný zápis copánku ten nejkratší možný.

Druhou obtížnou úlohou odvozenou od *word problem* je takzvaný *membership problem* nebo též *zobecněný word problem*. Pro danou podgrupu $H \subseteq B_n$ je úkolem rozhodnout, zda daný copánek $b \in B_n$ náleží do H. Dle (Ko a Potapov, 2017) je tento problém pro B_3 NP-úplný, a už pro B_5 dokonce nerozhodnutelný.

2.2 Conjugacy problem

Další zásadní úloha na copánkových grupách je *conjugacy problem*. Z kryptografického hlediska je tento problém obzvlášť významný, jelikož je na něm (nebo na problémech z něj odvozených) založena většina kryptosystémů. Proto se mu v této práci budeme věnovat důkladněji než předchozím problémům.

Máme dvě verze tohoto problému, a to *conjugacy decision problem* a *conjugacy search problem* (CSP). Přestože CSP je pro kryptografii zásadnější, zaměříme se nejdřív na *conjugacy decision problem* a na konci kapitoly si ukážeme, jak z jeho řešení snadno odvodit i řešení CSP.

Definice 9. *Nechť $b, c \in B_n$. Řekneme, že copánek c je* konjugovaný *s copánkem b, pokud $\exists x \in B_n \colon c = x^{-1}bx$. Pro konjugované copánky píšeme $b \sim c$.*

Pozorování. $\sim$ je relací ekvivalence na B_n .

Conjugacy decision problem je rozhodovací úloha, v níž dostaneme dva copánky a máme určit, zda jsou konjugované.

2.2.1 Summit set

První řešení *conjugacy decision problem* nabídl (Garside, 1969). V této práci si ukážeme důkaz jeho správnosti, který bude principiálně velmi podobný Garsidovu důkazu, ale obejde se bez definic týkajících se Cayleyho diagramu.

Připomeňme, že pro copánek v normální formě $b = \Delta_n^u A_1 \cdots A_p$ nazýváme u mocninou copánku b.

Značení. Mocninu copánku b budeme značit $m(b)$.

Definice 10. *Mějme copánek $b \in B_n \colon b = \sigma_{j_1}^{m_1} \cdots \sigma_{j_k}^{m_k}$, kde $j_i \in \{1, \ldots, n-1\}$ a $m_i \in \{1, -1\}$. Pak součet $\sum_{i=1}^{k} m_i$ nazveme* indexovou délkou *copánku.*

Pozorování. Indexová délka copánku nezávisí na konkrétní formě zápisu. Navíc pokud má $x \in B_n$ indexovou délku t, tak x^{-1} má indexovou délku $-t$. Tudíž pokud $b \sim c$ ($b = x^{-1}cx$), pak b a c mají stejnou indexovou délku.

Nyní si dokážeme dvě lemmata, pomocí nichž následně zkonstruujeme užitečnou konečnou množinu copánků, které jsou konjugované s daným copánkem. Nejdřív však zopakujeme poznámku učiněnou po zavedení normální formy.

Připomeňme, že copánek nazveme pozitivním, pokud jej lze popsat slovem bez negativních křížení σ_i^{-1}. Množinu pozitivních copánků značíme B_n^+.

Poznámka. V některých důkazech budeme normální formu psát jako $b = \Delta_n^{m(b)} B$ pro $B \in B_n^+$, jelikož nebude nutné pracovat s konkrétní podobou B.

Lemma 11. *Nechť $t, m \in \mathbb{Z}$, pak existuje jen konečný počet copánků b s indexovou délkou t a $m(b) \geq m$.*

Důkaz. Označme indexovou délku fundamentálního copánku Δ_n písmenem d. Uvažujme libovolný copánek b splňující podmínky v lemmatu, tj. má indexovou délku t a $m(b) \geq m$. Převeďme b do normální formy $b = \Delta_n^{m(b)} B$ pro nějaký pozitivní copánek $B \in B_n^+$ s indexovou délkou $l \geq 0$. Pak platí

$$t = m(b)d + l.$$

Protože $l \geq 0$, tak jistě $m(b) \leq \frac{t}{d}$, a dále víme, že $l = t - m(b)d$. Jelikož máme konečný počet generátorů, pak i možných $B \in B_n^+$ s indexovou délkou právě l je konečně mnoho. Dohromady tedy máme konečně možností na mocninu b, jelikož $m \leq m(b) \leq \frac{t}{d}$, a konečně pozitivních copánků B, které mohou následovat po $\Delta_n^{m(b)}$.

□

Lemma 12. *Je-li $b \sim c$, pak existuje pozitivní copánek $X \in B_n^+$, že $b = X^{-1}cX$.*

Důkaz. Máme $b = x^{-1}cx$ pro nějaké $x \in B_n$. Vyjádříme x v normální formě $x = \Delta_n^m X$ pro $X \in B_n^+$ a dostaneme $b = X^{-1}\Delta_n^{-m} c \Delta_n^m X$. Nyní rozlišíme dva případy podle parity m.

- m sudé: Použijeme důsledek lemmatu 2, který říká $\forall b \in B_n \colon b\Delta_n^2 = \Delta_n^2 b$. Proto máme
$$b = X^{-1}c\Delta_n^{-m}\Delta_n^m X = X^{-1}cX.$$
- m liché:
$$b = X^{-1}\Delta_n^{-1}\Delta_n^{-(m-1)} c \Delta_n^{m-1}\Delta_n X = X^{-1}\Delta_n^{-1} c \Delta_n X,$$
kde $\Delta_n X$ je jistě pozitivní copánek.

□

Využitím předchozích dvou lemmat můžeme zkonstruovat užitečnou konečnou množinu konjugovaných prvků k danému copánku b, a to takzvaný *summit set*. Nejdřív jej zadefinujeme formálně a následně pomocí algoritmu popíšeme jeho konstrukci převzatou od (Garside, 1969).

Definice 11. *Pro copánek $b \in B_n$* mocninou summit setu *nazýváme hodnotu*

$$m(SS(b)) = \max\{m(c) \mid c \sim b\}.$$

Summit set *prvku b je množina*

$$SS(b) = \{c \in B_n \mid c \sim b \ \wedge \ m(c) = m(SS(b))\}.$$

Před popsáním algoritmu připomeňme pojem permutační copánek. Pozitivní copánek nazveme permutační, pokud je možné jej nakreslit jako copánek, v němž se každé dva provázky kříží nanejvýš jednou. Množinu permutačních copánků značíme $\tilde{S_n}$ a permutační copánek indukující permutaci $\pi \in S_n$ značíme $\phi^{-1}(\pi)$.

Algoritmus 2 Konstrukce summit setu

Vstup: $b \in B_n$
Výstup: $SS(b)$

$S^{(0)} \leftarrow \{b\}$

$S^{(1)} \leftarrow \{c \in B_n \mid (\exists A \in \tilde{S_n} \colon c = A^{-1}bA) \ \wedge \ (m(c) \geq m(b))\}$

$i \leftarrow 1$

while $|S^{(i)}| > |S^{(i-1)}|$ **do**

$S^{(i+1)} \leftarrow \{\}$

for $c \in S^{(i)}$ **do**

$S^{(i+1)} \leftarrow S^{(i+1)} \cup \{d \in B_n \mid (\exists A \in \tilde{S_n} \colon d = A^{-1}cA) \ \wedge \ (m(d) \geq m(b))\}$

$i \leftarrow i + 1$

$k \leftarrow \max\{m(c) \mid c \in S^{(i)}\}$

$SS(b) \leftarrow \{c \in S^{(i)} \mid m(c) = k\}$

return $SS(b)$

Již jsme si rozmysleli, že konjugované prvky mají stejnou indexovou délku, a proto lemma 11 (o konečném počtu copánků s danou indexovou délkou a mocninou větší nebo rovnou než $m \in \mathbb{Z}$) zajišťuje konečnost této konstrukce. Lemma 12 navíc říká $\forall a \sim b \ \exists X \in B_n^+ \colon a = X^{-1}bX$, tudíž opakovaným konjugováním permutačními copánky, jejichž složením můžeme získat libovolný pozitivní copánek, dostaneme veškeré možné konjugace.

Nyní se pokusíme dokázat zásadní větu říkající, že pro copánky $b, c \in B_n$ platí:

$$b \sim c \iff SS(b) = SS(c).$$

K dokázání této věty budeme potřebovat několik technických lemmat. Nejdřív si připomeňme pojmy začínající a koncová množina. Pro $A \in B_n^+$ máme:

$$S(A) = \{i \mid \exists A' \in B_n^+ \colon A = \sigma_i A'\},$$
$$F(A) = \{i \mid \exists A' \in B_n^+ \colon A = A'\sigma_i\}.$$

Dále připomeňme lemma 8 a jeho důsledek, které popisují podobou začínající

a koncové množiny u permutačních copánků.

Pro permutační copánek $B = \phi^{-1}(\pi)$ jsou následující tvrzení ekvivalentní

(1) $i \in S(B)$,

(2) provázky i a $i+1$ se kříží v B,

(3) $\pi(i+1) < \pi(i)$.

Dále nechť k-tý provázek končí v B na j-té pozici a l-tý provázek na $(j+1)$-ní pozici, tj. $\pi(k) = j, \pi(l) = j+1$. Pak jsou následující tvrzení ekvivalentní

(1) $j \in F(B)$,

(2) k-tý a l-tý provázek se kříží v B,

(3) $k > l$.

Lemma 13. *Uvažujme permutační copánky $B, C \in \tilde{S}_n$ splňující $BC = \Delta_n$. Pak platí $F(B) \cup S(C) = \{1, \ldots, n-1\}$ a toto sjednocení je disjunktní.*

Důkaz. Vzpomeňme si, že v Δ_n se všechny provázky kříží právě jednou, a to při nakreslení jakéhokoliv pozitivního slova, které Δ_n popisuje. Kdyby existovalo $i \in S(C) \cap F(B)$, tak můžeme psát $\Delta_n = B'\sigma_i\sigma_i C'$ pro $B', C' \in B_n^+$, a tedy by se jeden pár provázků křížil přinejmenším dvakrát.

Dále uvažujme $i \notin F(B)$, pak chceme dokázat $i \in S(C)$. Nechť $B = \phi^{-1}(\rho)$ a položme $k = \rho^{-1}(i)$, $l = \rho^{-1}(i+1)$. Z důsledku lemmatu 8 máme, že l-tý a k-tý provázek se v B nekříží. Ale v $BC = \Delta_n$ se všechny páry provázků kříží, tudíž se k-tý provázek (provázek, který je na začátku C na i-té pozici) a l-tý provázek (provázek, který je na začátku C na $(i+1)$-ní pozici) musí křížit v C. Pak dle lemmatu 8 $i \in S(C)$.

□

Tvrzení 17, které bude brzy následovat, zobecňuje lemma 13 pro pozitivní copánky obsahující Δ_n, k čemuž potřebujeme zavést relaci obsahování.

Definice 12. *Pro pozitivní copánky $a, b \in B_n^+$ píšeme $a \preceq b$ a říkáme b* obsahuje *a, pokud $\exists c \in B_n^+ : ac = b$.*

Pozorování. Relace $\preceq$ je částečné uspořádání na B_n .

Zobecnění lemmatu 13 se může zdát jako snadný důsledek, jeho důkaz je však značně komplikovaný. Důkaz od (Garside, 1969) je velmi technický, my se tak pokusíme jít jinou cestou. K tomu však budeme potřebovat několik pomocných lemmat.

Lemma 14. *Mějme copánek* $b = A_1A_2\cdots A_p$, *kde* $\forall i \in \{1,\ldots,p\}\colon A_i \in \tilde{S_n}\backslash\{\Delta_n\}$ *a* $\forall i \in \{1,\ldots,p-1\}\colon S(A_{i+1}) \subseteq F(A_i)$. *Pak* $S(A_1) = S(b)$.

Důkaz. Jistě $S(A_1) \subseteq S(b)$, tedy zbývá $S(b) \subseteq S(A_1)$. Nejdřív si všimneme, že $b = A_1A_2\cdots A_p$ je v normální formě. Nyní pro spor uvažujme $i \in S(b)\backslash S(A_1)$, pak můžeme psát $b = \sigma_i b'$ pro nějaké $b' \in B_n^+$. Nyní na b spustíme Garsideův–Thurstonův algoritmus, tedy jej převedeme do normální formy. V algoritmu nejdřív rozdělíme b na nějaké permutační copánky $b = A'_1A'_2\cdots A'_q$ a následně A'_1 nanejvýš prodlužujeme. Kvůli jednoznačnosti normální formy (věta 7) nakonec dostaneme $b = A_1A_2\cdots A_p$, a tudíž máme $S(A'_1) \subseteq S(A_1)$. Ale $i \in S(A'_1)$, a tedy $i \in S(A_1)$, což je spor. Proto $S(b) \subseteq S(A_1)$.

□

Před dokázáním dalšího pomocného lemmatu připomeňme lemma 6, které říká, že každý permutační copánek lze zleva i zprava doplnit na fundamentální copánek, tj. $\forall B \in \tilde{S_n}\colon (\exists C_1 \in \tilde{S_n}\colon BC_1 = \Delta_n) \wedge (\exists C_2 \in \tilde{S_n}\colon C_2B = \Delta_n)$.

Lemma 15. *Pro permutační copánek* $B \in \tilde{S_n}$ *platí*

$$B = \Delta_n \iff S(B) = \{1,\ldots,n-1\}.$$

Důkaz. " $\Longrightarrow$ ": Plyne z různých slov popisujících Δ_n v lemmatu 1.

" $\Longleftarrow$ ": Dle lemmatu 6 lze každý permutační copánek zleva doplnit na Δ_n permutačním copánkem, tj. existuje $C \in \tilde{S_n}\colon CB = \Delta_n$. Následně z lemmatu 13 dostáváme $F(C) \cup S(B) = \{1,\ldots,n-1\}$ a toto sjednocení je disjunktní. Ale z předpokladu $S(B) = \{1,\ldots,n-1\}$, tudíž $F(C) = \{\}$, čili $C = \varepsilon$ a $B = \Delta_n$.

□

Lemma 16. *Mějme copánek* $b \in B_n^+$, *že* $S(b) = \{1,\ldots,n-1\}$. *Pak platí* $\Delta_n \preceq b$

Důkaz. Vezměme b splňující $S(b) = \{1,\ldots,n-1\}$ a vyjádřeme jej v normální formě

$$b = \Delta_n^{m(b)}A_1\cdots A_p,$$

kde $\forall i\colon A_i \in \tilde{S_n}\backslash\{\Delta_n\}$.

Pro spor předpokládejme, že $m(b) = 0$. Použitím lemmatu 14 dostáváme $S(A_1) = S(b) = \{1,\ldots,n-1\}$, kde druhá rovnost platí z předpokladu. Z lemmatu 15 pak máme $A_1 = \Delta_n$, ale přitom $A_1 \in \tilde{S_n}\backslash\{\Delta_n\}$, což je spor. Tudíž $m(b) > 0$, a proto $\Delta_n \preceq b$.

□

Důsledek. Pokud $F(b) = \{1, \ldots, n-1\}$, pak platí $\Delta_n \preceq b$

Důkaz. Připomeňme, že pro $b = \sigma_{i_1} \cdots \sigma_{i_k}$ značíme $\pi_n(b) = \sigma_{n-i_1}\sigma_{n-i_2} \cdots \sigma_{n-i_k}$ a z důsledku lemmatu 2 platí $b\Delta_n = \Delta_n \pi_n(b)$.

Rozmyslíme si, že předchozí lemma platí pro copánek s obráceným pořadím generátorů. Definujme $\mathscr{R}\colon B_n \to B_n$, že $\mathscr{R}(\sigma_{i_1}^{m_1} \cdots \sigma_{i_k}^{m_k}) = \sigma_{i_k}^{m_k}\sigma_{i_{k-1}}^{m_{k-1}} \cdots \sigma_{i_1}^{m_1}$, tj. zobrazení obracející pořadí generátorů.

Uvědomíme si, že $F(b) = S(\mathscr{R}(b))$, a aplikací předchozího lemmatu tak dostáváme $\Delta_n \preceq \mathscr{R}(b)$. Máme tedy $\mathscr{R}(b) = \Delta_n b'$ pro $b' \in B_n^+$, a tudíž $b = \mathscr{R}(b')\mathscr{R}(\Delta_n)$. Navíc $\Delta_n = \mathscr{R}(\Delta_n)$, což plyne z definujícího slova fundamentálního copánku (definice 3) $\Delta_n = (\sigma_1 \cdots \sigma_{n-1}) \cdots (\sigma_1\sigma_2)(\sigma_1)$ a slova $\Delta_n = (\sigma_1)(\sigma_2\sigma_1) \cdots (\sigma_{n-1} \cdots \sigma_1)$ z lemmatu 1. Použitím lemmatu 2 získáme $b = \Delta_n \pi_n(\mathscr{R}(b'))$, kde $\pi_n(\mathscr{R}(b')) \in B_n^+$, čili $\Delta_n \preceq b$.

□

Tvrzení 17. *Nechť $a, b \in B_n^+\colon \Delta_n \preceq ab$, pak*

$$F(a) \cup S(b) = \{1, \ldots, n-1\}.$$

Důkaz. Začneme převedením copánku b do normální formy $b = \Delta_n^u B_1 \cdots B_p$, kde

$$\forall i \in \{1, \ldots, p-1\}\colon S(B_{i+1}) \subseteq F(B_i). \tag{2.1}$$

Kdyby $u > 0$, tak máme $\{1, \ldots, n-1\} = S(\Delta_n) \subseteq S(b)$, kde první rovnost platí z lemmatu 15. Tedy $S(b) = \{1, \ldots, n-1\}$ a věta platí. Dále uvažujeme

$$b = B_1 \cdots B_p.$$

Budeme postupovat indukcí podle p.
$\underline{p = 1}$: Dle lemmatu 6 můžeme copánek $b = B_1 \in \tilde{S}_n$ doplnit zleva na fundamentální copánek permutačním copánkem, tj. $\exists C \in \tilde{S}_n\colon Cb = \Delta_n$. Z předpokladu $\Delta_n \preceq ab$, pak využitím důsledku lemmatu 2 máme

$$ab = \Delta_n d = \pi_n(d)\Delta_n = \pi_n(d)Cb \implies a = \pi_n(d)C \implies F(C) \subseteq F(a),$$

pro nějaké $d \in B_n^+$. Lemma 13 nám pak dává $F(C) \cup S(b) = \{1, \ldots, n-1\}$, což implikuje $F(a) \cup S(b) = \{1, \ldots, n-1\}$.
$\underline{p > 1}$: Máme $ab = aB_1B_2 \cdots B_p$ a označme $X = aB_1 \cdots B_{p-1}$. Dle indukčního předpokladu pro $p = 1$ platí $F(X) \cup S(B_p) = \{1, \ldots, n-1\}$. Ale dle (2.1)

$$S(B_p) \subseteq F(B_{p-1}) \subseteq F(X) \implies F(X) = \{1, \ldots, n-1\}.$$

Použitím důsledku lemmatu 16 dostáváme $\Delta_n \preceq X$. Použijeme indukční předpoklad pro $p-1$ pro copánky a a $B_1 \cdots B_{p-1}$, tedy máme

$$F(a) \cup S(B_1 \cdots B_{p-1}) = \{1, \ldots, n-1\} \implies$$
$$\implies F(a) \cup S(B_1 \cdots B_{p-1} B_p) = \{1, \ldots, n-1\}.$$

□

Zavedeme ještě další tři pomocná lemmata, která se již týkají konjugace. V jejich důkazu budeme postupovat podobně jako (Garside, 1969).

Lemma 18. *Mějme $h, d \in B_n$ takové, že $d = Y^{-1}hY$ pro nějaké $Y \in B_n^+$. Jistě existují pozitivní copánky $D, H \in B_n^+$: $h = \Delta_n^u H, d = \Delta_n^v D$ (nemusí nutně platit $u = m(h), v = m(d)$). Je-li $v > u$, tak platí $\Delta_n \preceq HY$.*

Důkaz. Rozepíšeme konjugaci ve znění lemmatu

$$\begin{aligned} d &= Y^{-1}hY, \\ \Delta_n^v D &= Y^{-1}\Delta_n^u HY, \\ \Delta_n^{-u} Y \Delta_n^v D &= HY, \\ \Delta_n^{v-u} Y' D &= HY, \end{aligned}$$

kde

$$Y' = \begin{cases} Y, & v \text{ sudé} \\ \pi_n(Y), & v \text{ liché} \end{cases}$$

a tedy $Y' \in B_n^+$. Jelikož $v > u$, tak máme $\Delta_n \preceq HY$.

□

Lemma 19. *Mějme copánky $b, c \in B_n$ a $X \in B_n^+$ splňující:*

- *$c \in SS(b)$, $c = \Delta_n^u C$, kde $u = m(c)$ a $C \in B_n^+$,*
- *$X = AY$ pro $A \in \tilde{S}_n$: $S(Y) \subseteq F(A)$,*
- *$v = m(X^{-1}cX) \geq u$.*

Pak platí $A^{-1}cA \in SS(b)$.

Důkaz. Z definice summit setu copánek $d \in B_n$: $d \sim b$ náleží do $SS(b)$, právě když $m(d) = m(SS(b))$ $(= \max\{m(c) \mid c \sim b\})$. Jelikož z předpokladu $c \in SS(b)$, tak $u = m(SS(b))$, a tudíž pro získání $A^{-1}cA \in SS(b)$ potřebujeme ověřit, že platí $m(A^{-1}cA) = u$. Máme

$$X^{-1}cX = Y^{-1}A^{-1}cAY \tag{2.2}$$

a mocnina tohoto prvku je z předpokladu $v \geq u$. Z lemmatu 6 můžeme A doplnit na Δ_n permutačním copánkem, tj. $\exists D \in \tilde{S}_n \colon AD = \Delta_n$. Pak

$$A^{-1}cA = A^{-1}\Delta_n^u CA = A^{-1}AD\Delta_n^{u-1}CA = \Delta_n^{u-1}D'CA, \tag{2.3}$$

kde

$$D' = \begin{cases} D, & u \text{ liché} \\ \pi_n(D). & u \text{ sudé} \end{cases}$$

Nyní můžeme využít lemma 18 na konjugaci (2.2) s tím, že h ze znění lemmatu bude $h = A^{-1}cA \overset{(2.3)}{=} \Delta_n^{u-1}D'CA$. Předpoklad je splněn, protože $v \geq u > u - 1$, a tedy dostáváme

$$\Delta_n \preceq D'CAY.$$

Nyní nám tvrzení 17 dává

$$F(D'CA) \cup S(Y) = \{1, \ldots, n-1\}.$$

Dále z předpokladu máme $S(Y) \subseteq F(A) \subseteq F(D'CA)$, a proto

$$F(D'CA) = \{1, \ldots, n-1\}.$$

Z důsledku lemmatu 16 tak máme $\Delta_n \preceq D'CA$. Dále

$$\Delta_n \preceq D'CA \implies m(\Delta_n^{u-1}D'CA) \geq u \overset{(2.3)}{\implies} m(A^{-1}cA) \geq u.$$

Navíc jistě $m(A^{-1}cA) \leq u$, jelikož $u = m(SS(b))$, a proto máme požadované $m(A^{-1}cA) = u$.

□

Lemma 20. *Pro $b, c \in B_n$ platí*

$$SS(b) = SS(c) \iff \exists X \in SS(b) \cap SS(c).$$

Důkaz. *"$\implies$": Platí triviálně.*

"$\impliedby$": Vezměme libovolné $B \in SS(b)$. Máme $B \sim X \sim c$, a navíc platí $m(B) = m(X) = m(SS(c))$, tedy jistě $B \in SS(c)$. Tudíž $SS(b) \subseteq SS(c)$ a analogicky dostaneme $SS(c) \subseteq SS(b)$.

□

Nyní konečně máme prostředky pro dokázání zásadní věty popisující vztah mezi konjugací a summit sety.

Věta 21. *Pro copánky $b, c \in B_n$ platí:*

$$b \sim c \iff SS(b) = SS(c).$$

Důkaz. " $\Longleftarrow$ ": Vezměme $X \in SS(b)$ $(= SS(c))$. Pak $b \sim X \sim c$, tedy $b \sim c$.

" $\Longrightarrow$ ": Vezměme libovolné $B \in SS(b), C \in SS(c)$. BÚNO $m(C) \geq m(B)$. Z předpokladu dostaneme $B \sim C$, a lemma 12 nám tak dává existenci $X \in B_n^+$ takového, že $C = X^{-1}BX$. Převedeme X do normální formy

$$X = \Delta_n^m A_1 A_2 \cdots A_p,$$

kde navíc můžeme uvažovat $m \in \{0, 1\}$, jelikož $X^{-1}\Delta_n^{-2k} B \Delta_n^{2k} X = X^{-1}BX$.

Nejdřív uvažujme $m = 0$. Pak máme složení copánků $A_i \in \tilde{S_n} \backslash \{\Delta_n\}$, kde pro každou dvojici $A_i A_{i+1}$ platí $S(A_{i+1}) \subseteq F(A_i)$. Označme $Y = A_2 \cdots A_p$, pak použitím lemmatu 14 dostaneme $S(Y) = S(A_2)$, tudíž $S(Y) \subseteq F(A_1)$. Nyní máme splněné všechny předpoklady lemmatu 19, z nějž pak dostaneme $A_1^{-1} B A_1 \in SS(b)$. Induktivně získáme $C \in SS(b)$ a z lemmatu 20 máme $SS(b) = SS(c)$.

Pro $m = 1$ položme $Y = A_1 \cdots A_p$. Jistě $S(Y) \subseteq F(\Delta_n) = \{1, \ldots, n-1\}$, a tedy použitím lemmatu 19 dostaneme $\Delta_n^{-1} B \Delta_n \in SS(B)$. Dále můžeme pokračovat analogicky jako v případu $m = 0$.

□

Věta nám přímo dává první způsob řešení *conjugacy decision problem*. Pro dva copánky $b, c \in B_n$ určíme množiny $SS(b)$ a $SS(c)$. Pokud $SS(b) = SS(c)$, pak $b \sim c$ a pokud ne, tak b a c konjugované nejsou.

Je poměrně snadné si rozmyslet, že obzvlášť pro velká n je algoritmus 2 (konstrukce summit setu) časově, a především prostorově náročný. První zlepšení nabídl hned (Garside, 1969). Uvažujme copánek $b \notin SS(b)$, pak jistě konjugováním s permutačními copánky musíme získat $c \sim b$ takové, že $m(c) > m(b)$. Z věty 21 plyne, že $SS(b) = SS(c)$. Můžeme tedy konstruovat summit set pro c, čili v množinách $S^{(i)}$ z algoritmu 2 nemusíme uvažovat copánky s mocninou menší než $m(c)$.

Další zlepšení Garsideova původního řešení souvisí s lemmatem 20, které nám říká, že pro porovnání dvou summit setů prvků b, c nám stačí určit $SS(b)$ a libovolný prvek $C \in SS(c)$. Pak $SS(b) = SS(c) \iff C \in SS(b)$. Problém je, že pro získání prvku z $SS(c)$ je třeba nejdřív určit mocninu summit setu, k čemuž jsme dosud museli spočíst celý $SS(c)$. (Elrifai a Morton, 1994) však nabídli alternativní způsob k nalezení prvku v summit setu, a to pomocí takzvaného *cyklení*.

Definice 13. Cyklení *je funkce* $c\colon B_n \to B_n$, *že pro* $b \in B_n$ *v normální formě* $b = \Delta_n^r A_1 A_2 \cdots A_p$ *je* $c(b) = \Delta_n^r A_2 A_3 \cdots A_p A_1'$, *kde* $A_1' = \begin{cases} A_1, & r \text{ sudé} \\ \pi_n(A_1). & r \text{ liché} \end{cases}$

Pozorování. Pro b a $c(b)$ z předchozí definice máme $c(b) = A_1'^{-1} b A_1'$, tedy $b \sim c(b)$.

Pozorování. $c(b)$ není obecně v normální formě a tedy máme $m(c(b)) \geq m(b)$

Z předchozích dvou pozorování a lemmatu 11 o konečném počtu určitých konjugovaných prvků dostáváme, že opakovanou aplikací cyklení nakonec začneme

získávat stejné prvky. (Elrifai a Morton, 1994) ukázali, že takto dokonce získáme prvek náležící summit setu. Tento proces lze provést v polynomiálním čase.

Spojením obou zmíněných zlepšeních můžeme kompletně obejít konstrukci popsanou v algoritmu 2 (konstrukce summit setu). Navíc můžeme využít pozorování, že konjugované copánky mají stejnou indexovou délku, tj. součet exponentů generátorů. Základní řešení *conjugacy decision problem* zapíšeme v podobě algoritmu.

Algoritmus 3 řešení *conjugacy decision problem*

Vstup: $b, c \in B_n$

Výstup: $\begin{cases} \text{TRUE} & b \sim c \\ \text{FALSE} & b \not\sim c \end{cases}$

(1) Pokud mají b a c různou indexovou délku, **return** FALSE.

(2) Pomocí cyklení získáme prvky $B \in SS(b)$ a $C \in SS(c)$.

(3) Pomocí algoritmu 2 budujeme $SS(B)$ $(= SS(b))$ dokud

(a) nezjistíme, že $C \in SS(B) \rightarrow$ **return** TRUE,

(b) nevybudujeme celý $SS(B) \rightarrow$ **return** FALSE.

Pro dlouhé copánky, a především pak pro velká n stále však toto řešení zůstává v praxi nepoužitelné. Problematická je především velikost samotného summit setu. Je však možné jej zmenšit a stále zachovat platnost věty 21.

2.2.2 Super summit set

Nejdřív zobecníme relaci $\preceq$ i mimo pozitivní copánky. Připomínáme, že pro $B, C \in B_n^+$: $B \preceq C$ pokud $\exists A \in B_n^+ : C = BA$, tj. pokud C obsahuje B. Ačkoliv značení zůstává pro obecné copánky stejné, již nemá tak očividnou geometrickou interpretaci.

Definice 14. *Pro $b, c \in B_n$ píšeme $b \preceq c$ pokud $\exists A \in B_n^+ : c = bA$*

Značení. Pro $b \in B_n$ píšeme $b \in [r, s]$ pokud $\Delta_n^r \preceq b \preceq \Delta_n^s$

Definice 15. *Pro $b \in B_n$ definujeme* infimum *a* supremum *copánku jako hodnoty*

$$\inf(b) = \max\{u \in \mathbb{Z} \mid \Delta_n^u \preceq b\},$$
$$\sup(b) = \min\{k \in \mathbb{Z} \mid b \preceq \Delta_n^k\}.$$

Dále definujeme kanonickou délku *copánku b jako*

$$\ell(b) = \sup(b) - \inf(b)$$

Pozorování. Pro $b \in B_n$ platí $m(b) = \inf(b)$

Normální forma nám dává poměrně dobrou představu i o supremu prvku. Využitím důsledku lemmatu 6 o doplnění permutačního copánku na fundamentální dostáváme následující pozorování.

Pozorování. Pro copánek v normální formě $b = \Delta_n^r A_1 \cdots A_p$ platí $\sup(b) \leq r + p$.

Lze ukázat dokonce rovnost, ale pro následující konstrukce není takové tvrzení zvláště přínosné. Připomeňme, že summit set copánku b je množina prvků konjugovaný s b s největší možnou mocninou. Konstrukci můžeme vidět v algoritmu 2.

(Elrifai a Morton, 1994) nabídli poměrně přirozené vylepšení summit setu, kde se krom maximalizování infima snažíme minimalizovat supremum.

Definice 16. *Pro copánek $b \in B_n$ uvažujeme hodnotu*

$$\ell(SSS(b)) = \min\{\ell(c) \mid c \sim b\}.$$

Super summit set *prvku b je množina*

$$SSS(b) = \{c \in B_n \mid c \sim b \ \wedge \ \ell(c) = \ell(SSS(b))\}$$

Konstrukce super summit setu copánku b je podobná algoritmu 2 s tím, že na začátku uvažujeme interval $[r^{(0)}, s^{(0)}] = [\inf(b), \sup(b)]$. Pokud v průběhu najdeme konjugovaný prvek $c \sim b$, který náleží do menšího intervalu $[\inf(c), \sup(c)]$, tak pokládáme $[r^{(1)}, s^{(1)}] = [\inf(c), \sup(c)]$. Takto pokračujeme dále a zahazujeme copánky $d \sim b$: $d \notin [r^{(i)}, s^{(i)}]$. Stejně jako u summit setu je díky lemmatu 11 tento proces konečný. Oproti summit setu je však jistě časově i prostorově efektivnější.

(Elrifai a Morton, 1994) dokázali, že platí obdoba věty 21, tedy

$$b \sim c \iff SSS(b) = SSS(c).$$

Pro jejich další užitečný výsledek připomeňme pojem cyklení. Pro copánek v normální formě $b = \Delta_n^r A_1 A_2 \cdots A_p$ je cyklení $c(b) = \Delta_n^r A_2 A_3 \cdots A_p A_1'$, kde $A_1' = A_1$ je-li r sudé a jinak $A_1' = \pi_n(A_1)$. (Elrifai a Morton, 1994) ukázali, že pomocí cyklení a takzvaného *decyklení* lze najít jeden prvek ze super summit setu v polynomiálním čase.

2.2.3 Ultra summit set

S dalším zmenšením super summit setu přišel (Gebhardt, 2003). Zavedl takzvaný *ultra summit setu*, pro nějž stále platí obdoba věty 21, tedy prvky jsou konjugované, právě když mají stejný ultra summit set.

Definice 17. *Pro copánek $b \in B_n$ definujeme* ultra summit set *jako speciální podmnožinu super summit setu:*

$$USS(b) = \{d \in SSS(b) \mid \exists k \in \mathbb{N}\colon c^k(d) = d\}$$

Na první pohled by se mohlo zdát, že podmínka $c^k(d) = d$ pro nějaké $k \in \mathbb{N}$ je splněna z podstaty cyklení. Problémem je, že pro $d \in B_n$ v normální formě, tj. $d = \Delta_n^r A_1 A_2 \cdots A_p$, obvykle $c(d) = \Delta_n^r A_2 A_3 \cdots A_p A_1'$ není v normální formě. Tedy obecně nemusí platit $c^2(b) = \Delta_n^r A_3 \cdots A_p A_1' A_2'$, $c^3(b) = \Delta_n^r A_4 \cdots A_p A_1' A_2' A_3'$, a tak dále. Ve skutečnosti, jak ukázal (Gebhardt, 2003), je USS výrazně menší než SSS, který se pro grupy s $n > 5$ stává takřka nepoužitelným.

K nalezení ultra summit setu je třeba nejdřív najít jeden jeho prvek. K tomu nám pomůže následující pozorování, které dostáváme okamžitě z definice.

Pozorování. Pro $b \in [r, s]$ platí $c(b) \in [r, s]$. Tedy máme

$$b \in SSS(a) \implies c(b) \in SSS(a).$$

Máme-li prvek $B \in SSS(b)$, pak opakovaným cyklením získáme $c^i(B) = c^j(B)$, protože super summit set je konečný. Tedy $c^i(B) \in USS(b)$. Hodnotu $i \in \mathbb{N}$, tedy počet nutných cyklení než získáme prvek z ultra summit setu, se pro obecný copánek zatím nepodařilo odhadnout.

(Gebhardt, 2003) ukázal, že vybudování ultra summit setu probíhá v čase

$$\mathcal{O}(|USS(b)|p + q),$$

kde p je polynom v $|b|$, tedy v délce b v artinovských generátorech, a q je časová složitost nalezení prvního prvku v $USS(b)$. Přestože na q nemáme horní odhad, empiricky jde o zanedbatelnou hodnotu.

Následující tabulka převzatá od (Gebhardt, 2003) nabízí srovnání USS a SSS. Obsahuje průměrné hodnoty $|USS(b)|$, $|SSS(b)|$, tj. počet prvků obou množin, kde „—“ píšeme pokud velikost přesáhla paměť 512MB. Tabulka také nabízí průměrné doby výpočtů t_S a t_U pro náhodné copánky tvaru $b = \delta_n^k A_1 A_2 \cdots A_r$, kde δ_n je fundamentální copánek pro *band generators* (viz sekce 2.1.2) a A_i jsou takzvané *simple elements*, které tvoří speciální podmnožinu permutačních copánků.

$n = 6$						
r	2	5	10	20	100	1000
$\lvert USS(b)\rvert$	15	17	21	40	200	2000
$\lvert SSS(b)\rvert$	270	3800	1.1e4	—	—	—
t_U	1.3ms	1.9ms	1.6ms	3.1ms	53ms	5.2s
t_S	18ms	600ms	24s	—	—	—

Velikost super summit setu i doba jeho výpočtu roste s n exponenciálně. Naopak velikost ultra summit setu zůstává pro pozorovaná n konstantní a u časové složitosti výpočtu byl zaznamenán lineární růst. Konkrétní hodnoty pro $n = 100$ máme v následující tabulce.

$n = 100$						
r	2	5	10	20	100	1000
$\|USS(b)\|$	15	17	21	40	200	2000
t_U	20ms	27ms	36ms	49ms	210ms	23s

Průměrně jsou ultra summit sety oproti předchozím množinám malé, dokonce dle (Garber, 2007) pro náhodný copánek obvykle platí $|USS(b)| = 2r$. Existují však případy malých copánků s velkým ultra summit setem, jimiž se zabývá například (Prasolov, 2009).

2.2.4 Conjugacy search problem

Krom rozhodovací verze *conjugacy problem*, kterou jsme se zabývali dosud, existuje i takzvaný *conjugacy search problem*(CSP). Úlohou je pro dva konjugované copánky $b \sim c$ najít copánek $x \in B_n \colon c = x^{-1}bx$.

Tento problém je pro kryptografii výrazně zásadnější než rozhodovací verze, jak si ukážeme v následující kapitole. Naštěstí jej umíme řešit analogickým postupem jako *conjugacy decision problem*, jen si musíme ukládat prvky, kterými konjugujeme. Na další straně si ukážeme úpravu algoritmu 3 (řešení *conjugacy decision problem*).

Algoritmus 4 řešení *conjugacy search problem*

Vstup: $b, c \in B_n : b \sim c$
Výstup: $x \in B_n : c = x^{-1}bx$

(1) Pomocí cyklení získáme prvky $B^{(0)} \in SS(b)$ a $Y \in B_n^+ : B^{(0)} = Y^{-1}bY$. Analogicky $C \in SS(c)(= SS(b))$ a $Z \in B_n^+ : C = Z^{-1}cZ$.

(2) Pomocí algoritmu 2 budujeme $SS(B)$ a pro každý nový prvek $B^{(i)} \in SS(B)$ ukládáme

- $X^{(i)} \in \tilde{S}_n : B^{(i)} = (X^{(i)})^{-1}B^{(j)}X^{(i)}$, pro nějaké $j < i$,
- odkaz na $B^{(j)}$ z předchozí bodu, tedy „předchůdce“ $B^{(i)}$.

Skončíme jakmile $B^{(i)} = C$.

(3) Zpětnou propagací najdeme $X = X^{(i_1)}X^{(i_2)} \cdots X^{(i_k)}$ takové, že $C = X^{-1}BX$.

(4) **return** YXZ^{-1}

Algoritmus jistě skončí, protože $C \in SS(b)$. Zbývá si rozmyslet, že algoritmus skutečně vrací, to co chceme. Máme

$$\begin{aligned} B &= Y^{-1}bY, \\ C &= X^{-1}BX, \\ c &= ZCZ^{-1}, \\ \implies c &= (ZX^{-1}Y^{-1})b(YXZ^{-1}). \end{aligned}$$

Navíc si všimneme, že ke každému prvku summit setu ukládáme navíc pouze permutační copánek. Paměťová náročnost tak vzroste minimálně, a navíc víme, že $b \sim c$, tedy takřka nikdy nemusíme budovat celý $SS(b)$.

Analogicky lze rozšířit algoritmy využívající super summit set a ultra summit set. Máme tak efektivní řešení CSP(*conjugacy search problem*).

Poznámka. $x^{-1}bx = x'^{-1}bx'$ neimplikuje, že $x = x'$. Řešení CSP tedy není jednoznačné.

3. Kryptosystémy na copánkových grupách

V této kapitole si představíme několik kryptosystémů pracujících nad copánkovými grupami a následně si ukážeme možné způsoby, jak popsaná schémata prolomit. Na závěr se zamyslíme nad kryptografickým využitím problémů, které nejsou odvozené od *conjugacy problem*. Konkrétně se zaměříme na *membership problem* a nabídneme možný způsob, jak jej využít při konstrukci kryptografických schémat.

3.1 Kryptosystémy

3.1.1 Diffieho–Hellmanova výměna klíčů

Dohoda na klíči nebo též *výměna klíčů* je metoda domluvení tajného symetrického klíče přes veřejný kanál, kde má útočník přístup ke všem zprávám. Pomocí domluveného klíče pak mohou účastníci tajně komunikovat využitím symetrického šifrování.

Jeden z nejznámějších asymetrických protokolů je Diffieho–Hellmanova výměna klíčů, kterou je možné provést i nad copánkovou grupou. Než si představíme copánkovou verzi, popíšeme pomocí obrázku původní schéma navržené (Diffie a Hellman, 1976), které pracuje nad konečnou grupou $\mathbb{Z}_p^*$. Šipky znázorňují posílání zprávy přes veřejný kanál.

Veřejné parametry:

p, g, kde p je prvočíslo a g generátor grupy $\mathbb{Z}_p^*$

Alice		**Bob**
$a \in \mathbb{N}$		$b \in \mathbb{N}$
$A = x^a$		$B = x^b$
	$\xrightarrow{\quad A \quad}$	
	$\xleftarrow{\quad B \quad}$	
$s = B^a$		$s = A^b$

Oba účastníci se zjevně domluvili na stejném klíči s, protože $g^{ab} = g^{ba}$ a pro prolomení schématu musí být útočník schopen řešit diskrétní logaritmus.

Než si ukážeme úpravu pro copánkové grupy, zavedeme užitečné značení a potřebnou definici.

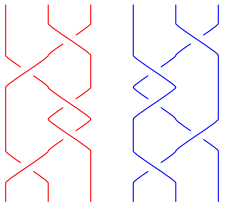
Značení. Pro copánky $b, x \in B_n$ píšeme $b^x = x^{-1}bx$.

Definice 18. *Pro $n \in \mathbb{N}$ a $m \approx n/2$ definujeme* levou třídu copánků *jako podgrupu*

$$B^L_{n,m} \subseteq B_n \colon B^L_{n,m} = \langle \sigma_1, \sigma_2, \ldots, \sigma_{m-1} \rangle.$$

Analogicky definujeme pravou třídu copánků

$$B^P_{n,m} \subseteq B_n \colon B^P_{n,m} = \langle \sigma_{m+1}, \sigma_{m+2}, \ldots, \sigma_{n-1} \rangle.$$



Obrázek 3.1: příklady copánků z $B^L_{6,3}$ a $B^P_{6,3}$

Poznámka. Neplatí, že $b \in B^L_{n,m} \iff b$ je zapsaný pomocí generátorů $B^L_{n,m}$. Například pro $n = 4, m = 2$ máme $\sigma_1\sigma_2\sigma_2^{-1} \in B^L_{n,m} = \langle \sigma_1 \rangle$.

Pozorování. Prvky z $B^L_{n,m}$ a $B^P_{n,m}$ spolu komutují, tj.

$$\forall a \in B^L_{n,m}\ \forall b \in B^P_{n,m} \colon ab = ba.$$

Pozorování. Pokud $b \in B^L_{n,m}$, pak $b^{-1} \in B^L_{n,m}$. Analogicky pro $B^P_{n,m}$.

Nyní již můžeme zavést Diffieho–Hellmanovu výměnu klíčů nad copánkovou grupou.

Veřejné parametry:

$n \in \mathbb{N}$,

$m \approx n/2$,

$x \in B_n \backslash (B^L_{n,m} \cup B^P_{n,m})$

Alice		**Bob**
$a \in B^L_{n,m}$		$b \in B^P_{n,m}$
$A = x^a$		$B = x^b$
	$\xrightarrow{\quad A \quad}$	
	$\xleftarrow{\quad B \quad}$	
$s = B^a$		$s = A^b$

Díky komutaci platí

$$B^a = (x^b)^a = a^{-1}b^{-1}xba = b^{-1}a^{-1}xab = (x^a)^b = A^b,$$

tedy účastníci skutečně spočetli stejný klíč.

(Garber, 2007) doporučuje bezpečností parametr $n = 80$ a hodnota pro copánek $b = \Delta_n^{m(b)} A_1 \cdots A_p$, má být hodnota p rovna 12.

Rovnou zmíníme první slabinu schématu, a to takzvaný *man-in-the-middle* útok. Ten se týká obecné verze protokolu, nijak nesouvisí s copánky. Podstatou je, že útočník zachytí vyměňované zprávy a domluví si dva různé klíče s oběma účastníky, aby se za ně pak mohl v komunikaci vydávat. Útok ukazuje následující schéma:

Alice	**Eva**	**Bob**
$a \in B_{n,m}^L$	$a' \in B_{n,m}^L \ \wedge \ b' \in B_{n,m}^P$	$b \in B_{n,m}^P$
$A = x^a$	$A' = x^{a'} \ \wedge \ B' = x^{a'}$	$B = x^b$
	$A \longrightarrow$	$A' \longrightarrow$
	$\longleftarrow B'$	$\longleftarrow B$
$s_1 = (B')^a$	$s_1 = A^{b'} \ \wedge \ s_2 = B^{a'}$	$s_2 = (A')^b$

Eva se nyní při komunikaci s Alicí může vydávat za Boba a naopak. Jako účinná ochrana proti tomuto útoku slouží autentizační schéma, které je možné navrhnout i na copánkových grupách, jak ukázal např. (Dehornoy, 2006).

3.1.2 Autentizační schéma

V následujícím schématu chce Alice navázat komunikaci s Bobem, který však nejdřív potřebuje ověřit její identitu. Konkrétně pravá Alice vlastní pár (s, v), kde s je soukromý klíč a v veřejný klíč. Bob chce pomocí v ověřit, zda údajná Alice (osoba, která s ním navazuje kontakt) skutečně vlastní s. Než autentizační schéma popíšeme, zadefinujeme si novou operaci na copánkových grupách. Začněme potřebným značením.

Značení. Funkci posunutí, tj. zvýšení všech indexů generátorů o jedna, budeme značit $d\colon B_n \to B_{n+1}$. Máme tedy

$$d(\sigma_{i_1}^{m_1}\sigma_{i_2}^{m_2}\cdots\sigma_{i_k}^{m_k}) = \sigma_{i_1+1}^{m_1}\sigma_{i_2+1}^{m_2}\cdots\sigma_{i_k+1}^{m_k}.$$

Značení. Běžné skládání copánků $a, b \in B_n$ budeme v této sekci psát jako $\cdot$, tedy $a \cdot b = ab$.

Definice 19. Posunutou konjugaci *definujeme jako funkci* $* : B_n \times B_n \to B_{n+1}$, *která pro* $x, b \in B_n$ *dává*

$$*(x, b) = x \cdot d(b) \cdot \sigma_1 \cdot d(x^{-1}).$$

Značení. Posunutou konjugaci budeme psát jako operaci, tj. $x * b = *(x, b)$.

Kryptografie založená na posunuté konjugaci závisí na složitosti *shifted conjugacy search problem*: máme $v, v' \in B_n$, kde $v' = s * v$ pro nějaké $s \in B_n$, pak je výpočetně náročné určit $s' \in B_n$: $v' = s' * v$. Toho (Dehornoy, 2006) využil k vytvoření následujícího autentizačního schématu.

Alicin soukromý klíč:
$s \in B_n$
Veřejné parametry:
$n \in \mathbb{N},\ v \in B_n,\ v' = s * v$
Protokol:

(1) Alice zvolí náhodné $r \in B_n$ a pošle Bobovi závazek $x = r * v$ a $x' = r * v'$.

(2) Bob zvolí náhodný bit $c \in \{0, 1\}$ a pošle jej Alici.

(3) Je-li $c = 0$, pak Alice pošle $y = r$. Bob ověří, zda $y * v = x$ a $y * v' = x'$.
Je-li $c = 1$, pak Alice pošle $y = r * s$. Bob ověří, zda $y * x = x'$.

Zajímat nás bude především bod (3), v němž Bob ověřuje, zda Alice skutečně zná tajný klíč s. Je-li $c = 0$, pak Alice prokáže, že poslala korektní závazek, tedy že nezvolila taková x, x', aby pak mohla falšovat znalost s. Brzy si ukážeme, že je-li $c = 1$, pak Alice prokazuje, že vlastní soukromý klíč s. Jelikož hodnotu bitu c volí Bob, musí být Alice schopná prokázat obě možnosti výše. Navíc Bob není schopen zjistit Alicin soukromý klíč (kdyby znal r i $r * s$, je schopen snadno spočíst s).

V praxi se schéma popsané výše opakuje k-krát pro bezpečnostní parametr $k \in \mathbb{N}$, jinak by útočník (tedy falešná Alice) mohl správně uhádnout s pravděpodobností $1/2$, jaké c Bob pošle a zvolit adekvátní závazek. S k opakováními je tak pravděpodobnost úspěchu při opakovaném hádání 2^{-k}.

Nyní si důkladněji rozmyslíme, jak probíhá případ $c = 1$, v němž Alice prokazuje znalost soukromého klíče s, aniž by jej odhalila Bobovi. Potřebnou vlastnost posunuté konjugace zformulujeme jako lemma, k jehož důkazu využijeme následující snadná pozorování týkající se posunutí.

Pozorování. Pro $a, b \in B_n$ platí

- $d(a \cdot b) = d(a) \cdot d(b)$,

- $d(a^{-1}) = d(a)^{-1}$,
- $\sigma_1 \cdot d^2(a) = d^2(a) \cdot \sigma_1$.

Lemma 22. *Posunutá konjugace je zleva samodistributivní, tj. platí*

$$\forall a, b, c \in B_n \colon a * (b * c) = (a * b) * (a * c).$$

Důkaz. Rozepíšeme pravou stranu:

$$\begin{aligned}
&(a * b) * (a * c) = (a * b) \cdot d(a * c) \cdot \sigma_1 \cdot d(a * b)^{-1} \\
&= (a \cdot d(b) \cdot \sigma_1 \cdot d(a)^{-1}) \cdot d(a \cdot d(c) \cdot \sigma_1 \cdot d(a)^{-1}) \cdot \sigma_1 \cdot d(d(a) \cdot \sigma_1^{-1} \cdot d(b)^{-1} \cdot a^{-1}) \\
&= a \cdot d(b) \cdot \sigma_1 \cdot d^2(c) \cdot \sigma_2 \cdot d^2(a)^{-1} \cdot \sigma_1 \cdot d^2(a) \cdot \sigma_2^{-1} \cdot d^2(b)^{-1} \cdot d(a)^{-1} \\
&= a \cdot d(b) \cdot d^2(c) \cdot \sigma_1 \cdot \sigma_2 \cdot \sigma_1 \cdot \sigma_2^{-1} \cdot d^2(b)^{-1} \cdot d(a)^{-1} \\
&= a \cdot d(b) \cdot d^2(c) \cdot \sigma_2 \cdot \sigma_1 \cdot d^2(b)^{-1} \cdot d(a)^{-1} \\
&= a \cdot d(b) \cdot d^2(c) \cdot d(\sigma_1) \cdot d^2(b)^{-1} \cdot \sigma_1 \cdot d(a)^{-1} \\
&= a \cdot d(b \cdot d(c) \cdot \sigma_1 \cdot d(b)^{-1}) \cdot \sigma_1 \cdot d(a)^{-1} \\
&= a * (b * c).
\end{aligned}$$

□

V autentizačním schématu pak platí

$$y * x = (r * s) * (r * v) \overset{L22}{=} r * (s * v) = r * v' = x',$$

tudíž Alice v případě $c = 1$ prokáže znalost soukromého klíče s a Bob by musel být schopen řešit *shifted conjugacy search problem*, aby soukromý klíč odhalil.

V tomto schématu jsme si ukázali, že běžnou konjugaci můžeme pro potenciální zvýšení bezpečnosti nějak upravit. Existuje několik dalších podobných variant, my si však místo nich ukážeme jiný způsob, jak učinit běžný CSP (*conjugacy search problem*) složitějším.

3.1.3 Commutator protocol

Pro pochopení principu následujícího schématu je zásadní pozorování, které jsme již učinil v předchozí kapitole.

Pozorování. Řešení CSP není jednoznačné.

Problém tak můžeme útočníkovi zkomplikovat zmenšením množiny všech řešení. Tohoto dosahuje úloha *multiple simultaneous conjugacy search problems*, jejíž podstatou je hledání řešení více rovnic najednou. Konkrétně dostaneme dvě k-tice copánků $(a_1, a_2, \ldots, a_k)$ a $(x^{-1}a_1x, x^{-1}a_2x, \ldots, x^{-1}a_kx)$ pro nějaké $x \in B_n$ a úkolem je najít $x' \in B_n$ takové, že splňuje $\forall i \in \{1, \ldots, k\}\colon x'^{-1}a_ix' = x^{-1}a_ix$.

(Anshel a kol., 2001b) na této úloze založili takzvaný *commutator protocol*, který zajišťuje domluvu na tajném klíči.

Veřejné klíče:
$S = \langle s_1, s_2, \ldots, s_k \rangle$, kde $s_i \in B_n$
$T = \langle t_1, t_2, \ldots, t_l \rangle$, kde $t_i \in B_n$
Protokol:

(1) Alice zvolí $a = s_{i_1}s_{i_2}\cdots s_{i_u}$ a pošle Bobovi dvojice

$$\{(t_1, at_1a^{-1}), (t_2, at_2a^{-1}), \ldots, (t_l, at_la^{-1})\}.$$

Bob zvolí $b = t_{j_1}t_{j_2}\cdots t_{j_v}$ a pošle Alici dvojice

$$\{(s_1, b^{-1}s_1b), (s_2, b^{-1}s_2b), \ldots, (s_k, b^{-1}s_kb)\}.$$

(2) Alice spočte

$$(b^{-1}s_{i_1}b)(b^{-1}s_{i_2}b)\cdots(b^{-1}s_{i_u}b)a^{-1} = (b^{-1}ab)a^{-1}.$$

Bob spočte

$$b^{-1}(at_{j_1}a^{-1})(at_{j_2}a^{-1})\cdots(at_{j_v}a^{-1}) = b^{-1}(aba^{-1}).$$

Účastníci se takto dohodli na tajném klíči $b^{-1}aba^{-1}$. Pro jeho prolomení musí útočník vyřešit MSCSP (*multiple simultaneous conjugacy search problems*).

3.2 Útoky na kryptosystémy

3.2.1 Útok pomocí ultra summit setu

Všimněme si, že všechny tři popsané kryptosystémy jsou založené na konjugaci, či na její variaci. Zdánlivě tak k jejich prolomení stačí vyřešit CSP, čehož jsme dosáhli v předchozí kapitole nabídnutím algoritmu, který pracoval se summit setem. Navíc jsme zavedli podmnožinu summit setu, a to takzvaný ultra summit set (USS), s nímž lze CSP vyřešit výrazně efektivněji. Stručně nastíníme princip tohoto řešení:

Algoritmus 5 řešení CSP pomocí ultra summit setu

Vstup: $b, c \in B_n : b \sim c$
Výstup: $d \in B_n : c = d^{-1}bd$

(1) Pomocí cyklení a decyklení najdi $B, C \in USS(b)\,(= USS(c))$ a zapamatuj si konjugující copánky $x, y \in B_n : B = x^{-1}bx$, $C = y^{-1}cy$.

(2) Konstruuj prvky $USS(B)\,(= USS(b))$ dokud nenajdeš prvek C a zapamatuj si $z \in B_n : C = z^{-1}Bz$.

(3) **return** xzy^{-1}

Důvodem, proč tento algoritmus nerozbíjí triviálně všechna schémata využívající konjugaci, je nejednoznačnost řešení CSP. Konkrétně spustíme-li algoritmus na prvky $a \sim x^{-1}ax$, dostaneme řešení x', pro které obecně nemusí platit $x' = x$. Vlastnosti, které má x, tak x' nemusí sdílet. Ukážeme si však tvrzení, které dává hrubý návod, jak z libovolného řešení x' zkonstruujeme řešení s požadovanými vlastnostmi. Před formulací tvrzení zavedeme potřebný pojem.

Definice 20. *Pro $b \in B_n$ nazveme množinu všech prvků, které komutují s b,* centralizér *prvku b. Značíme jej $C(b)$.*

Pozorování. Pro $b \in B_n$ platí $C(b) = \{c \in B_n \,|\, cb = bc\} = \{c \in B_n \,|\, b = c^{-1}bc\}$, a navíc je $C(b)$ podgrupa B_n .

Tvrzení 23. *Nechť $x \in B_n$ řeší CSP pro copánky $a \sim b$, tj. $a = x^{-1}bx$. Pak*

$$\textit{copánek } cx \textit{ řeší CSP pro } a \sim b \iff c \in C(b).$$

Důkaz.

- " $\Longleftarrow$ " : Máme $c \in C(b)$, tedy $c^{-1}bc = b$. Proto $x^{-1}c^{-1}bcx = x^{-1}bx = a$.
- " $\Longrightarrow$ " : Máme $x^{-1}c^{-1}bcx = a = x^{-1}bx \implies c^{-1}bc = b \implies c \in C(b)$.

□

Jak si brzy ukážeme, pro žádný z kryptosystémů popsaných v této práci nestačí k jejich prolomení najít jen libovolné řešení CSP. Postupně tak projdeme úpravy útoku pro konkrétní schémata a popíšeme jejich úspěšnost.

Útok na Diffieho–Hellmanův protokol

Vzpomeneme si, že v Diffieho–Hellmanově výměně klíčů má útočník k dispozici copánky x, $x^a = a^{-1}xa$ a $x^b = b^{-1}xb$, kde

$$a \in B^L_{n,m} = \langle \sigma_1, \sigma_2, \ldots, \sigma_{m-1} \rangle,$$
$$b \in B^P_{n,m} = \langle \sigma_{m+1}, \sigma_{m+2}, \ldots, \sigma_{n-1} \rangle,$$

pro $n \in \mathbb{N}$ a $m \approx n/2$. Tajný klíč je pak $x^{ab} = x^{ba}$, kde rovnost platí, protože spolu a a b komutují.

Vyřešením CSP pro $x \sim x^a$ najdeme nějaké $a' \in B_n \colon x^{a'} = x^a$. Může však nastat $a' \notin B^L_{n,m}$, a tedy $x^{ba'} \neq x^{ba}$. Použitím tvrzení 23 víme, že existuje copánek $c \in C(x) \colon ca' \in B^L_{n,m}$. Nabízí se tak možnost postupně zkoušet prvky z centralizéru, k čemuž nejdřív musíme určit jeho generátory, což je ovšem obecně náročné.

Existují polynomiální algoritmy pro nalezení centralizéru určitých třídy copánků, ale pro obecný copánek dosud není známá efektivní metoda výpočtu. Funkční algoritmus nabídl např. (Franco a López, 2003), ale u jeho řešení se očekává exponenciální složitost.

(Gebhardt, 2006) však ukázal, že pro prolomení Diffieho–Hellmanovy výměny klíčů není třeba konstruovat celý centralizér. Dokonce je postačující následující pozorování.

Pozorování. Pro $x \in B_n$ platí $\langle \Delta_n^2, x \rangle \subseteq C(x)$.

Algoritmus 6 Útok na Diffieho-Hellmanovu výměnu klíčů

Vstup: x, x^a, x^b, kde $a \in B^L_{n,m}, b \in B^P_{n,m}$, pro $n \in \mathbb{N}, m \approx n/2$
Výstup: x^{ab}

(1) Pomocí algoritmu 5 (řešení CSP) najdi $c \in B_n \colon x^c = x^a$.

(2) **for** $i \in \{-5, \ldots, 5\}$ **do**

(a) Polož $c' = x^i c$.

(b) Ověř následující dvě podmínky:

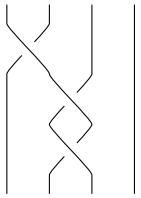
- pro permutaci $\pi_{c'}$ indukovanou c' platí $\pi_{c'} \restriction_{\{m+1,\ldots,n-1\}} = id$, tj. permutace nijak neprohazuje pravé provázky,
- pro $y = \Delta_n^{-m(c')} c'$ platí, že $S(y) \subseteq \{1, \ldots, m-1\}$, tj. y může začínat pouze kříženími z levé třídy copánků.

Pokud obě podmínky platí, **return** $(x^b)^{c'}$.

Je zřejmé, že algoritmus nemusí najít řešení, protože snadno může nastat $\langle \Delta_n^2, x\rangle \neq C(x)$. Dokonce však může najít chybné řešení, jak si ukážeme na jednoduchém příkladu.

Příklad. Mějme $n = 4, m = 2$, proto $B_{n,m}^L = \langle \sigma_1 \rangle$ a $B_{n,m}^P = \langle \sigma_3 \rangle$. Položme $x = \sigma_2, a = \sigma_1^{-1}, b = \sigma_3, c = \sigma_1\sigma_2\sigma_2$, kde c máme na obrázku 3.2. Začneme upravením x^c, k čemuž využijeme relaci $\sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}$. Dostáváme

$$x^c = \sigma_2^{-1}\sigma_2^{-1}\sigma_1^{-1}\sigma_2\sigma_1\sigma_2\sigma_2 = \sigma_2^{-1}\sigma_2^{-1}\sigma_1^{-1}\sigma_1\sigma_2\sigma_1\sigma_2 = \sigma_2^{-1}\sigma_1\sigma_2.$$



Obrázek 3.2: $c = \sigma_1\sigma_2\sigma_2$

Nyní ověříme, že $x^c = x^a$:

$$\begin{aligned} \sigma_2^{-1}\sigma_1\sigma_2 &\stackrel{?}{=} \sigma_1\sigma_2\sigma_1^{-1} \\ \sigma_2^{-1}\sigma_1\sigma_2\sigma_1 &\stackrel{?}{=} \sigma_1\sigma_2 \\ \sigma_2^{-1}\sigma_2\sigma_1\sigma_2 &\stackrel{?}{=} \sigma_1\sigma_2 \\ \sigma_1\sigma_2 &= \sigma_1\sigma_2 \end{aligned}$$

Snadno si rozmyslíme, že c splňuje obě podmínky v algoritmu 6, proto by pro $i = 0$ algoritmus vrátil c jako správné řešení. Jenže máme $x^{bc} \neq x^{ba}$, což lze ověřit převodem copánku $(x^{bc})^{-1}x^{ba}$ do normální formy a jeho porovnáním s prázdnou posloupností ε.

Jistější způsob ověření, že c' komutuje s b, je ověřit komutaci se všemi generátory $B_{n,m}^P$. Na druhou stranu je tento proces časově náročnější, protože musíme $2n$-krát převést copánek do jeho normální formy. (Gebhardt, 2006) navíc empiricky ukázal, že podmínky v algoritmu 6 jsou postačující — ve $100,000$ pokusech získat tajný klíč nezaznamenal jediný neúspěch.

Útok na autentizační schéma

Vzpomeneme si, že autentizační schéma popsané v této práci využívá posunutou konjugaci $*$. Pro $x, b \in B_n$ máme

$$x * b = xd(b)\sigma_1 d(x^{-1}),$$

kde posunutí $d\colon B_n \to B_{n+1}$ je funkce zvyšující indexy všech generátorů o jedna.

Poznámka. Pro vstupy z B_n je výstup posunuté konjugace v B_{n+1}.

Shifted conjugacy search problem je úloha, v níž z copánků b a $x * b$ máme určit copánek $x' \in B_n\colon x' * b = x * b$. Zopakujeme schéma založené na tomto problému.

Alicin soukromý klíč:
$s \in B_n$
Veřejné parametry:
$n \in \mathbb{N},\ v \in B_n,\ v' = s * v$
Protokol:

(1) Alice zvolí náhodné $r \in B_n$ a pošle Bobovi závazek $x = r * v$ a $x' = r * v'$.

(2) Bob zvolí náhodný bit $c \in \{0, 1\}$ a pošle jej Alici.

(3) Je-li $c = 0$, pak Alice pošle $y = r$. Bob ověří, zda $y * v = x$ a $y * v' = x'$.
Je-li $c = 1$, pak Alice pošle $y = r * s$. Bob ověří, zda $y * x = x'$.

Cílem útočníka je určit soukromý klíč s, aby se mohl vydávat za Alici. Tedy potřebuje vyřešit *shifted conjugacy search problem* pro copánky v a $v' = s * v$.

Prvním krokem je převést úlohu na běžný CSP, k čemuž si vzpomeňme na copánek $\delta_n = \sigma_{n-1} \cdots \sigma_2 \sigma_1$ zavedený v sekci o *word problem* 2.1.2.

Lemma 24. *Pro $b \in B_n$ platí v grupě B_{n+1} rovnost*

$$d(b) = \delta_{n+1}^{-1} b \delta_{n+1}$$

Důkaz. Z relačních podmínek na copánkových grupách snadno dostaneme, že pro každé $i \in \{1, \ldots, n-1\}$ v grupě B_{n+1} platí $\sigma_i \delta_{n+1} = \delta_{n+1} \sigma_{i+1}$. Proto pak $b\delta_{n+1} = \delta_{n+1} d(b)$, a tedy $\delta_{n+1}^{-1} b \delta_{n+1} = d(b)$

□

Tvrzení 25. *Pro $v, s \in B_n$ v grupě B_{n+1} platí*

$$v' = s * v \iff v' \delta_{n+1}^{-1} = s d(v) \sigma_1 \delta_{n+1}^{-1} s^{-1}$$

Důkaz. Použijeme definici $*$ a lemma 24 na $d(s)$.

□

Útok nyní můžeme provést ve dvou krocích.

(1) Vyřešením CSP pro $v'\delta_{n+1}^{-1}$ a $d(v)\sigma_1\delta_{n+1}^{-1}$ najdi copánek $s' \in B_{n+1}$ splňující $v'\delta_{n+1}^{-1} = s' d(v) \sigma_1 \delta_{n+1}^{-1} s'^{-1}$.

(2) Najdi $c \in C(d(v)\sigma_1\delta_{n+1}^{-1})$ takové, že $t = s'c \in B_n$. Toto t řeší *shifted conjugacy search problem.*

Z tvrzení 23 (vztah mezi řešeními CSP) dostaneme existenci takového c a z tvrzení 25 máme, že t je skutečně hledané řešení. (Longrigg a Ushakov, 2008) však ukázali, že hledání vhodného $c \in C(d(v)\sigma_1\delta_{n+1}^{-1})$ je výrazně obtížnější než u Diffieho–Hellmanova schématu. K jeho nalezení využili heuristické prohledávání určité podmnožiny všech řešení (pro prohledaní množiny všech řešení by bylo třeba určit celý centralizér, což je obtížné). Heuristikou byla délka slova $(v * t')^{-1}v'$ v normální formě, kde $t' = s'c'$ pro $c' \in C(d(v)\sigma_1\delta_{n+1}^{-1})$ — čím kratší slovo bylo, tím blíže byli správnému řešení.

V experimentech (Longrigg a Ushakov, 2008) zaznamenali vysokou úspěšnost kroku (2) pro delší klíče ($|s| = 800$) nehledě na velikost n. Naopak pro kratší klíče ($|s| = 100$) ve většině případů klíč nezískali. Výsledky lze vysvětlit tím, že kratší copánky mají větší centralizér, a proto vybraná podmnožina centralizéru neobsahovala potřebné $c\colon t = s'c \in B_n$. Autoři navíc zmiňují, že zvolená heuristika může být špatná.

Z experimentů vyplývá, že pro dosažení vyšší bezpečnosti je vhodné volit kratší klíče. To sice umožňuje prostorově efektivnější implementaci, ale zároveň otvírá cestu brute-force útokům. Obecně tedy autentizační schéma založené na posunuté konjugaci není považováno za bezpečné.

Útok na commutator protocol

Vzpomeneme si, že commutator protocol je založený na složitosti MSCSP (*multiple simultaneous conjugacy search problems*), ve kterém pro dvě k-tice copánků $(a_1, a_2, \ldots, a_k)$ a $(a_1^x, a_2^x, \ldots, a_k^x)$ $(= (x^{-1}a_1x, \ldots, x^{-1}a_kx))$ hledáme $x' \in B_n$ takové, že $\forall i \in \{1, 2, \ldots, k\}\colon a_i^{x'} = a_i^x$. Rozmyslíme si, že útok pomocí řešení běžného CSP je značně náročný.

Pro jednoduchost uvažujeme dvě konjugace, tedy máme tajný klíč $x \in B_n$ a veřejné klíče a_1, a_2 a $b_1 = a_1^x, b_2 = a_2^x$. Útočník chce z veřejných klíčů určit tajný klíč vyřešením CSP. Pomocí algoritmu 5 (řešení CSP) může nalézt copánky $x_1, x_2 \in B_n\colon b_1 = a_1^{x_1}, b_2 = a_2^{x_2}$. Z tvrzení 23 (vztah mezi řešeními CSP) víme, že existují $c_1 \in C(a_1), c_2 \in C(a_2)\colon c_1x_1 = c_2x_2$. Tento copánek je řešením MSCSP.

Snadno si rozmyslíme, že určení copánků c_1, c_2 z předchozího odstavce je výrazně náročnější než u obou dosavadních útoků, jelikož musíme prohledávat dva centralizéry namísto jednoho. Pro více konjugací navíc složitost roste exponenciálně, a tudíž je útok pomocí ultra summit setu příliš výpočetně náročný. Ukážeme si však útok, který MSCSP řeší poměrně efektivně.

3.2.2 Délkový útok

Na rozdíl od útoku pomocí USS, k němuž je třeba obsáhlá teorie, je následující útok principiálně velmi jednoduchý. Takzvaný délkový útok (*length attack*)

navržený (Hughes a Tannenbaum, 2003) je pravděpodobností řešení MSCSP, jehož princip si popíšeme na běžném CSP, kde je útok teoreticky také aplikovatelný.

V této sekci budou všechny copánky generované prvky z generující množiny $D = \{s_1, \ldots, s_k, s_1^{-1}, \ldots, s_k^{-1}\}$, kde $s_i \in B_n$ (takto se většinou generují náhodné copánky, kde např. $D = \{s, s^{-1} | s \in \tilde{S_n}\}$). V CSP známe copánky a a a^x, kde $x = s_1 s_2 \cdots s_m$ s $s_i \in D$. Principem délkového útoku je postupné „odloupávání“ x z a^x — útočník zkouší konjugovat a^x s prvky D a dostává:

$$(a^x)^s = \begin{cases} s_{m-1}^{-1} \cdots s_1^{-1} a s_1 \cdots s_{m-1}, & s = s_m^{-1} \\ s^{-1} s_m^{-1} \cdots s_1^{-1} a s_1 \cdots s_m s. & s \in D \backslash \{s_m^{-1}\} \end{cases}$$

Pokud útočník konjugováním získá „kratší“ copánek, pak ví, že uhádl dobře. Tento proces opakuje, dokud nezíská a.

Mohlo by se zdát, že není-li $|D|$ příliš velké, tak tento útok triviálně prolomí veškeré kryptosystémy založené na konjugaci — nejen, že řeší CSP, dokonce okamžitě najde tajné x. Problém je v interpretování pojmu „kratší“, tedy ve vhodném zavedení délkové funkce $\ell \colon B_n \to \mathbb{N}$, jak si ukážeme v následujícím příkladu.

Příklad. Budeme pracovat v grupě B_4 s $D = \{\sigma_1, \sigma_2, \sigma_3, \sigma_1^{-1}, \sigma_2^{-1}, \sigma_3^{-1}\}$, tj. s běžnou generující množinou. Pro copánek $b \in B_4$ bude $\ell_{\Delta_n}(b)$ délka slova b v normální formě. Mějme $a = \sigma_1^{-1}\sigma_2\sigma_3\sigma_2^{-1}\sigma_1, x = \sigma_2\sigma_1$, pak

$$a^x = \sigma_1^{-1}\sigma_2^{-1}\sigma_1^{-1}\sigma_2\sigma_3\sigma_2^{-1}\sigma_1\sigma_2\sigma_1 = \sigma_2^{-1}\sigma_1^{-1}\sigma_2^{-1}\sigma_2\sigma_3\sigma_2^{-1}\sigma_2\sigma_1\sigma_2 \\ = \sigma_2^{-1}\sigma_1^{-1}\sigma_3\sigma_1\sigma_2 = \sigma_2^{-1}\sigma_3\sigma_2.$$

Při zkoušení všech generátorů dostaneme $(a^x)^{\sigma_2^{-1}} = \sigma_3$, zato pro správný generátor máme $(a^x)^{\sigma_1^{-1}} = \sigma_1\sigma_2^{-1}\sigma_3\sigma_2\sigma_1^{-1}$. Snadno ověříme, že $\ell_{\Delta_n}((a^x)^{\sigma_2^{-1}}) < \ell_{\Delta_n}((a^x)^{\sigma_1^{-1}})$, a navíc zřejmě není snadné nalézt rozumnou délkovou funkci ℓ', která by tuto situaci vyhodnotila opačně. Samozřejmě jde o degenerovaný případ, jelikož máme $\ell_{\Delta_n}(a^x) < \ell_{\Delta_n}(a)$, a proto copánek a^x nepotřebujeme zkracovat; situace, kdy konjugace se správným prvkem copánek prodlouží, však pro náhodné copánky skutečně mohou nastat.

V příkladu jsme zmínili jednu možnou délkovou funkci ℓ_{Δ_n}, tj. délku copánku v normální formě. Největším problémem této funkce jsou velké délky negativních křížení σ_i^{-1}. Vzpomeneme si totiž, že negativní křížení můžeme upravovat pomocí tvrzení 3, které říká $\forall i \in \{1, \ldots, n-1\}\, \exists A \in B_n^+ \colon\ \sigma_i^{-1} = \Delta_n^{-1} A$. Z podoby tohoto A (viz důkaz tvrzení 3) a z délky slovo Δ_n máme $\ell(\sigma_i^{-1}) = n(n-1) - 1$.

(Garber a kol., 2002) navrhli vhodnější délkovou funkci ℓ, která řeší tento problém. Využili toho, že dle lemmatu 6 můžeme každý permutační copánek zprava doplnit na fundamentální copánek, tj. $\forall A \in \tilde{S_n}\, \exists B \in \tilde{S_n} \colon AB = \Delta_n$. Toto lemma

přeformulujeme na $\forall A \in \tilde{S_n}\, \exists B \in \tilde{S_n}\colon \Delta_n^{-1} A = B^{-1}$. Pomocí tohoto vztahu můžeme copánek v normální formě $b = \Delta_n^{-r} A_1 A_2 \cdots A_p$ pro $r > 0$ přepsat na

$$b = \begin{cases} B_1^{-1} \cdots B_r^{-1} A_{r+1} \cdots A_p, & r < p \\ \Delta_n^{p-r} B_1^{-1} B_2^{-1} \cdots B_p^{-1}. & r \geq p \end{cases}$$

Délková funkce ℓ je pak délkou slova v této formě.

Další zlepšení délkového útoku spočívá v hádání více prvků z generující množiny D najednou, tedy porovnáváme délky copánků $(a^x)^{t_1 \cdots t_m}$ pro $t_i \in D$. Takto vyřešíme některé případy, kdy správný generátor copánek prodlužuje, jako tomu bylo v příkladu. Pochopitelně složitost tohoto postupu roste exponenciálně s m, což je problematické obzvlášť pro velkou $|D|$.

Na začátku sekce jsme zmínili, že délkový útok je vhodný především pro řešení MSCSP. Pro k-tici $(a_1^x, a_2^x, \ldots, a_k^x)$ můžeme totiž správnost odhadu $s \in D$ ověřit na více copáncích najednou. K tomu je třeba najít vhodné uspořádání na k-ticích $(\ell(a_1^{xs}), \ell(a_2^{xs}), \ldots, \ell(a_k^{xs}))$, čímž se zabývají například (Garber a kol., 2002).

Autoři v článku nabídli výsledky experimentů s délkovou funkcí ℓ_{Δ_n} a $m = 1$ (tedy hádali pouze po jednom generátoru). Dosáhli vysoké úspěšnosti zjištění x pro malá n, avšak už pro $n = 20$ dosáhli průměrně zhruba 15 úspěchů z 500 pokusů. Ve své další práci (Garber a kol., 2005) algoritmus vylepšili o možnost vrátit se o několik kroků zpět v případě zacyklení, čímž dosáhli výrazného zlepšení.

3.2.3 Útok pomocí reprezentací

Poslední útok, který zmíníme v této práci, je aplikovatelný na všechny tři popsané kryptosystémy. Podstatou je převod copánků na prvky lineární grupy, tj. na invertibilní matice nad nějakým okruhem R. V lineární grupě pak dle (Garber, 2007) existují snadné metody řešení CSP.

Homomorfismus z copánkové grupy do lineární grupy se nazývá *reprezentace*. V útoku na CSP na copánkových grupách postupujeme následovně:

(1) Zvol reprezentaci $\rho\colon B_n \to \mathrm{GL}(R)$ pro nějaký okruh R. Převeď copánky $a, b \in B_n\colon a \sim b$ na matice $A = \rho(a), B = \rho(b)$.

(2) Najdi matici $X \in \mathrm{GL}(R)\colon A = X^{-1} B X$.

(3) Urči vzor X v reprezentaci ρ, tj. takový copánek $x \in B_n\colon \rho(x) = X$. Pak platí $a = x^{-1} b x$ tedy x řeší CSP.

Krok (3) je obecně náročný, dokonce dle (Garber, 2007) takový obraz nemusí vždy nutně existovat, tedy nelze říct, že útok řeší CSP. Komplikované je také nalezení takového x, že splňuje určité podmínky (jak potřebujeme u všech popsaných kryptosystémů). Oba tyto problémy však vyžadují obsáhlejší teorii a jsou mimo záběr

této práce. Proto jen stručně představíme dvě reprezentace, které se na copánkových grupách používají, a to *Burauovu reprezentaci* a *Lawrencové–Krammerovu reprezentaci*. Oběma se v kryptografickém kontextu detailněji zaobírá např. (Garber, 2007).

Značení. $\mathbb{Z}[t^{\pm 1}]$ je okruh Laurentových polynomů s celočíselnými koeficienty, tj.

$$\mathbb{Z}[t^{\pm 1}] = \{a_k t^k + \cdots + a_m t^m \,|(\, k, m \in \mathbb{Z} \colon k \leq m) \;\wedge\; (\forall i \in \{k, \ldots, m\} \colon a_i \in \mathbb{Z})\}.$$

Definice 21. Burauova reprezentace *je homomorfismus* $\mathscr{B} \colon B_n \to \mathrm{GL}_n(\mathbb{Z}[t^{\pm 1}])$, *kde*

$$\mathscr{B}(\sigma_i) = \left(\begin{array}{c|cc|c} I_{i-1} & 0 & 0 & 0 \\ \hline 0 & 1-t & t & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & I_{n-i-1} \end{array}\right).$$

Pozorování. $\mathscr{B}(\sigma_i)$ skutečně náleží do $\mathrm{GL}_n(\mathbb{Z}[t^{\pm 1}]$, protože máme inverz

$$\mathscr{B}(\sigma_i)^{-1} = \left(\begin{array}{c|cc|c} I_{i-1} & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & t^{-1} & -t^{-1}+1 & 0 \\ \hline 0 & 0 & 0 & I_{n-i-1} \end{array}\right).$$

Navíc zřejmě platí $\mathscr{B}(\sigma_i) \cdot \mathscr{B}(\sigma_j) = \mathscr{B}(\sigma_j) \cdot \mathscr{B}(\sigma_i)$ pro $|i-j| > 1$ a roznásobením matic můžeme snadno ověřit $\mathscr{B}(\sigma_i) \cdot \mathscr{B}(\sigma_{i+1}) \cdot \mathscr{B}(\sigma_i) = \mathscr{B}(\sigma_{i+1}) \cdot \mathscr{B}(\sigma_i) \cdot \mathscr{B}(\sigma_{i+1})$. Tedy reprezentace zachovává podmínky určující copánkovou grupu.

Problém s Burauovou reprezentací je, že pro $n \geq 5$ není *věrná*, tj. nejde o prostý homomorfismus. To znamená, že určení vzoru matice může být nemožné, i když daná matice náleží do obrazu $\mathscr{B}$.

Tento problém řeší Lawrencové–Krammerova reprezentace, která je věrná, jak ukázal např. (Bigelow, 2000). Její definice je velmi technická a pro pochopení základního principu útoku není nijak zásadní. Využitím Lawrencové–Krammerovy reprezentace se zabývali např. (Cheon a Jun, 2003), kteří navrhli útok na Diffieho–Hellmanův protokol pracující v polynomiálním čase. Dle autorů by pro dosažení stejné bezpečnosti, jakou nabízí kryptografie na eliptické křivce P-521, muselo být $n = 10^5$ a délka domluveného tajného klíče pak zhruba $10^9 \approx 2^{30}$ bitů, tj. zhruba 130MB.

3.3 Další kryptografický směr na copánkových grupách

Všechny kryptosystémy na copánkových grupách popsané v této práci (i většina dalších) závisí buď přímo na složitosti *conjugacy search problem*, nebo na složitosti

souvisejících problémů. V této práci jsme si však ukázali řešení CSP, které je v průměru velmi rychlé, i jiné efektivní útoky na kryptosystémy založené na tomto problému. Nyní se tak zaměříme na jiný problém na copánkových grupách a nabídneme jeho možné využití při tvorbě kryptografických schémat.

Vzpomeneme si, že v sekci o *word problem* jsme zmínili takzvaný *membership problem* také nazývaný *zobecněný word problem*. V této úloze uvažujeme podgrupu $H \subseteq B_n$ a pro copánek $b \in B_n$ máme rozhodnout, zda $b \in H$. *Membership problem* je dle (Ko a Potapov, 2017) pro B_3 NP-těžký, a pro $B_n \colon n \geq 5$ dokonce nerozhodnutelný.

Pro navržení schématu potřebujeme „trapdoor“ funkci, tedy funkci, kterou je těžké invertovat bez znalosti určité další informace. Konkrétně pro *membership problem* chceme najít podgrupu $H \subseteq B_n$ takovou, že se znalostí nějakého tajného parametru p jsme schopni pro libovolné $b \in B_n$ efektivně rozhodnout, zda $b \in H$. Bez znalosti p pak tato úloha musí být výpočetně náročná, a navíc požadujeme, aby z H nebylo možné určit p. H pak bude veřejný klíč a p soukromý klíč.

V této práci navrhujeme využít jako $H \subseteq B_n$ centralizér nějakého copánku $b \in B_n$ s tím, že b bude tajný parametr, tj. $p = b$ a $H = C(b)$. Z definice pak libovolný copánek $c \in B_n$ náleží do $C(b)$, právě když komutuje s b, což můžeme ověřit v polynomiálním čase převodem copánků cb a bc do normálních forem a jejich porovnáním.

Nyní navrhneme, jak by mohlo vypadat schéma založené na této myšlence. Ve schématu Bob posílá Alici jeden bit informace.

---

(1) Alice zvolí $b \in B_n$ a spočte generátory $GC = \{g_1, \ldots, g_k\}$, že $\langle GC \rangle = C(b)$. Pak zveřejní GC.

(2) Bob zvolí zprávu $m \in \{0, 1\}$ a vytvoří šifrové slovo $y \in B_n$, kde

- pokud $m = 0$, pak $y = c_1 c_2 \cdots c_l$, kde $\forall i \in \{1, \ldots, l\} \colon c_i \in GC \cup GC^{-1}$ $(= \{g_1, \ldots, g_k, g_1^{-1}, \ldots, g_k^{-1}\})$, tedy $y \in C(b)$,
- pokud $m = 1$, pak $y = (c_1 c_2 \cdots c_{l_0}) a (c_{l_0+1} \cdots c_l)$, kde $c_i \in GC \cup GC^{-1}$ a $a \in B_n \backslash (GC \cup GC^{-1})$.

Pak y pošle Alici.

(3) Alice zjistí m následovně: $m = \begin{cases} 0, & y \in C(b) \\ 1. & y \notin C(b) \end{cases}$

V prvním kroku Alice určuje generátory centralizéru, což je pro obecný copánek náročné. Pro funkčnost schématu však nemusí určit celý centralizér, nýbrž jen jeho podgrupu, což je pochopitelně výrazně snazší. Tedy v prvním kroku Alice volí

GC, že $\langle GC\rangle \subseteq C(b)$. V takové případě je rozumné zveřejnit i malou množinu $A \subset B_n \backslash C(b)$, z níž pak Bob volí a.

Pro odhalení m musí být teoreticky útočník schopen řešit *membership problem*. Nemáme však obecnou podgrupu, nýbrž centralizér, což jednak může útoky zjednodušit, a jednak může útočník schéma prolomit zjištěním b z $\langle GC\rangle \subseteq C(b)$. Podívejme se proto na úpravu protokolu, která potenciální útok na b komplikuje.

---

(1) Alice zvolí b, $x \in B_n$. Následně spočte generátory $GC = \{g_1, \ldots, g_k\}$, že $\langle GC\rangle \subseteq C(b)$ a vezme množinu $A = \{a_1, \ldots a_r\} \subset B_n \backslash C(b)$. Pak zveřejní množiny $GC^x = \{x^{-1}gx,\ x^{-1}g^{-1}x \mid g \in GC\}$ a $A^x = \{x^{-1}ax \mid a \in A\}$.

(2) Bob zvolí zprávu $m \in \{0, 1\}$ a pošle Alici šifrové slovo $y \in B_n$, kde

- pokud $m = 0$, pak $y = c_1 c_2 \cdots c_l$, kde $c_i \in GC^x$,
- pokud $m = 1$, pak $y = (c_1 c_2 \cdots c_{l_0}) a (c_{l_0+1} \cdots c_l)$, kde $c_i \in GC^x$ a $a \in A^x$.

(3) Alice zjistí m následovně: $m = \begin{cases} 0, & xyx^{-1} \in C(b) \\ 1. & xyx^{-1} \notin C(b) \end{cases}$

Již jsme si rozmysleli, že u obou schémat potřebujeme pro dešifrování převést dva copánky do normální formy, což lze provést v čase $\mathcal{O}(|b|^2\, n \log n)$. Zato brute-force útok je exponenciálně složitý, konkrétně $\mathcal{O}((2|GC|)^l)$.

Tato schémata mají určité mezery, především není zřejmé, zda je *membership problem* náročný pro libovolnou $H \subseteq B_n$, či pouze pro určité podgrupy. Navíc volba copánku b značně ovlivní strukturu centralizéru (viz např. (Gebhardt, 2006) — diskuze v kapitole 4.1). Proto jde spíše o koncept, kudy by bylo možné dále směřovat design kryptografických schémat, a před aplikací by zasloužil důkladnější analýzu, která je ovšem mimo záběr této práce.

Závěr

V této práci jsme se po vybudování stručné teorie zabývali dvěma zásadními problémy na copánkových grupách, a to *word problem* a *conjugacy problem*. V poslední kapitole jsme následně představili několik kryptosystémů pracujících nad copánkovými grupami a ukázali principy možných útoků.

Soustředili jsme se především na obecné řešení CSP (*conjugacy search problem*), jelikož většina dosud navržených schémat je založená na konjugaci. V druhé kapitole jsme představili původní řešení a zmínili i algoritmus využívající ultra summit set, který již problém v průměru řeší velmi efektivně. Pro konkrétní kryptosystémy se však ukázalo, že řešit CSP je nedostatečné, protože jsme vždy potřebovali řešení splňující nějaké další podmínky. Pro prolomení schémat tak bylo třeba prohledávat prostor všech řešení charakterizovaný centralizérem nějakého copánku.

Popsali jsme i další útoky, a to délkový útok efektivní proti MSCSP (*multiple simultaneous conjugacy search problems*) a útok pomocí reprezentací, který například řeší Diffieho–Hellmanovu výměnu klíčů v polynomiálním čase.

Pro copánkové grupy bylo navrženo několik dalších protokolů, pro většinu byl však rychle nalezen efektivní útok — využívaný problém se podařilo převést na konjugaci, kterou pak efektivně řeší např. Lawrencové–Krammerova reprezentace, či na soustavu konjugací náchylných na délkový útok. Například však dosud nebyl popsán žádný útok na schéma využívají takzvaný *twisted conjugacy problem* navržené (Anshel a kol., 2001a), v praxi se ale přesto nevyužívá. Kvůli řešení CSP pomocí ultra summit setu a Lawrencové–Krammerově reprezentaci se totiž od žádných problému podobných konjugaci neočekává dostatečná bezpečnost.

Možnou naději pro kryptografii na copánkových grupách nabízejí dva problémy zmíněné v druhé kapitole, a to *shortest word problem* a *membership problem*. První úlohou je určit nejkratší zápis copánku a druhou rozhodnout, zda copánek náleží do podgrupy $H \subseteq B_n$. V této práci jsme se zamysleli nad možným využitím *membership problem* a navrhli případnou podobu schématu, které je na něm založené. Pro posouzení jeho bezpečnosti je třeba další výzkum, ale schéma přinejmenším dává naději, že copánkové grupy mají kryptografii ještě co nabídnout.

Seznam použité literatury

Anshel, I., Anshel, M., Fisher, B. a Goldfeld, D. (2001a). New key agreement protocols in braid group cryptography. volume 2020, pages 13–27. ISBN 978-3-540-41898-6. doi: 10.1007/3-540-45353-9_2.

Anshel, I., Anshel, M. a Goldfeld, D. (2001b). An algebraic method for public-key cryptography. *Mathematical Research Letters*, **6**. doi: 10.4310/MRL.1999.v6.n3.a3.

Artin, E. (1947). Theory of braids. *Annals of Mathematics*, **48**(1), 101–126. ISSN 0003486X. URL `http://www.jstor.org/stable/1969218`.

Artin, E. (1925). Theorie der zöpfe. URL `https://api.semanticscholar.org/CorpusID:124441713`.

Bigelow, S. J. (2000). Braid groups are linear.

Birman, J., Ko, K. H. a Lee, S. J. (1998). A new approach to the word and conjugacy problems in the braid groups. *Advances in Mathematics*, **139**(2), 322–353. ISSN 0001-8708. doi: https://doi.org/10.1006/aima.1998.1761. URL `https://www.sciencedirect.com/science/article/pii/S0001870898917613`.

Cheon, J. H. a Jun, B. (2003). A polynomial time algorithm for the braid diffie-hellman conjugacy problem. Cryptology ePrint Archive, Paper 2003/019. URL `https://eprint.iacr.org/2003/019`. `https://eprint.iacr.org/2003/019`.

Dehornoy, P. (1994). Braid groups and left distributive operations. *Transactions of the American Mathematical Society*, **345**, 115–150. URL `https://api.semanticscholar.org/CorpusID:18594612`.

Dehornoy, P. (1997). A fast method for comparing braids. *Advances in Mathematics*, **125**(2), 200–235. ISSN 0001-8708. doi: https://doi.org/10.1006/aima.1997.1605. URL `https://www.sciencedirect.com/science/article/pii/S0001870897916054`.

Dehornoy, P. (2006). Using shifted conjugacy in braid-based cryptography.

Diffie, W. a Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, **22**(6), 644–654. doi: 10.1109/TIT.1976.1055638.

Elrifai, E. a Morton, H. (1994). Algorithms for positive braids. *Quarterly Journal of Mathematics*, **45**. doi: 10.1093/qmath/45.4.479.

Franco, N. a López, J. (2003). Computation of centralizers in braid groups and garside groups. *Revista matemática iberoamericana, ISSN 0213-2230, Vol. 19, Nº 2, 2003, pags. 367-384*, **19**. doi: 10.4171/RMI/352.

Garber, D. (2007). Braid group cryptography. doi: 10.1142/9789814291415_0006.

Garber, D., Kaplan, S., Teicher, M., Tsaban, B. a Vishne, U. (2002). Length-based conjugacy search in the braid group. doi: 10.1090/conm/418/07947.

Garber, D., Kaplan, S., Teicher, M., Tsaban, B. a Vishne, U. (2005). Probabilistic solutions of equations in the braid group. *Advances in Applied Mathematics*, **35**(3), 323–334. ISSN 0196-8858. doi: 10.1016/j.aam.2005.03.002. URL http://dx.doi.org/10.1016/j.aam.2005.03.002.

Garside, F. A. (1969). THE BRAID GROUP AND OTHER GROUPS. *The Quarterly Journal of Mathematics*, **20**(1), 235–254. ISSN 0033-5606. doi: 10.1093/qmath/20.1.235. URL https://doi.org/10.1093/qmath/20.1.235.

Gebhardt, V. (2003). A new approach to the conjugacy problem in garside groups.

Gebhardt, V. (2006). Conjugacy search in braid groups. *Appl. Algebra Eng. Commun. Comput.*, **17**, 219–238. doi: 10.1007/s00200-006-0008-7.

Hughes, J. a Tannenbaum, A. (2003). Length-based attacks for certain group based encryption rewriting systems.

Ko, S.-K. a Potapov, I. (2017). Composition problems for braids: Membership, identity and freeness.

Longrigg, J. a Ushakov, A. (2008). Cryptanalysis of shifted conjugacy authentication protocol, arxiv preprint. *Journal of Mathematical Cryptology*, **2**. doi: 10.1515/JMC.2008.005.

Paterson, M. a Razborov, A. (1991). The set of minimal braids is co-np-complete. *Journal of Algorithms*, **12**(3), 393–408. ISSN 0196-6774. doi: https://doi.org/10.1016/0196-6774(91)90011-M. URL https://www.sciencedirect.com/science/article/pii/019667749190011M.

Prasolov, M. (2009). Small braids having a big ultra summit set.