SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Evidenčné číslo: FEI-5382-92202

AUTOMATICKÉ KOMPONOVANIE HUDBY PODĽA SPRÁVY (ŠIFROVANIE DO HUDBY)
BAKALÁRSKA PRÁCA

2020 **Pavol Sobota**

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Evidenčné číslo: FEI-5382-92202

AUTOMATICKÉ KOMPONOVANIE HUDBY PODĽA SPRÁVY (ŠIFROVANIE DO HUDBY)
BAKALÁRSKA PRÁCA

Študijný program :	Aplikovaná informatika
Číslo študijného odboru:	2511
Názov študijného odboru:	9.2.9 Aplikovaná informatika
Školiace pracovisko:	Ústav informatiky a matematiky
Vedúci záverečnej práce:	Ing. Peter Špaček
Konzultant ak bol určený:	Ing. Eugen Antal, PhD.

BRATISLAVA 2020 **Pavol Sobota**

Slovenská technická univerzita v Bratislave
Ústav informatiky a matematiky

Fakulta elektrotechniky a informatiky
Akademický rok: 2019/2020
Evidenčné číslo: FEI-5382-92202

ZADANIE BAKALÁRSKEJ PRÁCE

Študent:	**Pavol Sobota**
ID študenta:	92202
Študijný program:	aplikovaná informatika
Študijný odbor:	informatika
Vedúci práce:	Ing. Peter Špaček
Konzultant:	Ing. Eugen Antal
Miesto vypracovania:	Ústav informatiky a matematiky

Názov práce: **Automatické komponovanie hudby podľa správy (šifrovanie do hudby)**

Jazyk, v ktorom sa práca vypracuje: slovenský jazyk

Špecifikácia zadania:

Cieľom je vytvorenie systému na zakódovanie správy do hudby. Ide o kódovanie do melódie, nie do nahrávky. Komponovanie hudby bude spĺňať pravidlá pop-music (napr. 3, 4 akordy, stupnice dur a mol). Substituovať sa bude cez interval od tónu v aktuálnom akorde, nie od toniky, na výsledku nesmie byť poznať, že sa jedná o automatickú hudbu. Téma vyžaduje nie len vzdelanie v oblasti klasických šifier, ale hlavne hudobnej teórie. Pravidlá pre generovanie melódie budú závisieť práve od úrovne vzdelania v tejto oblasti. Vstupom by mala byť správa, výstupom MIDI (prípadne iný hudobný zápis). Úlohy:

1. Analyzovať techniky a metódy komponovania hudby.
2. Vytvoriť kódovací systém.
3. Vytvoriť softvérový nástroj na automatizované vytváranie komplexnejšej hudby zo správ (zohľadniť zmeny akordov, atď.).
4. Otestovať riešenie .

Zoznam odbornej literatúry:

1. Grošek, O. – Vojvoda, M. – Zajac, P. *Klasické šifry.* Bratislava : Vydavateľstvo STU, 2011. 214 s. ISBN 978-80-227-3486-8.
2. DAVID, Coelle. The Complete Guide to JFugue: Programming Music in Java. 2008. Dostupné tiež z: http://www.jfugue.org/4/jfbmrkklprpp/TheCompleteGuideToJFugue- v1.pdf.
3. Henke, Jamie. Basic Concepts of Music Theory. University of Wisconsin-Madison
4. Kelly, Sloan. (2019). Finite State Machines. 10.1007/978-1-4842-4533-0 _19.

Riešenie zadania práce od: 23. 09. 2019

Dátum odovzdania práce: 01. 06. 2020

Pavol Sobota
študent

Dr. rer. nat. Martin Drozda
vedúci pracoviska

prof. Dr. Ing. Miloš Oravec
garant študijného programu

SÚHRN

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Študijný program :	Aplikovaná informatika
Vyberte typ práce:	Automatické komponovanie hudby podľa správy (šifrovanie do hudby)
Autor:	Pavol Sobota
Vedúci záverečnej práce:	Ing. Peter Špaček
Konzultant ak bol určený:	Ing. Eugen Antal, PhD.
Miesto a rok predloženia práce:	Bratislava 2020

Cieľom je vytvorenie substitučnej šifry na zakódovanie správy do hudby. Do melódie, nie do nahrávky, nejedná sa o steganografiu. Komponovanie bude spĺňať pravidlá pop-music (napr. 3, 4 akordy, pentatonika, melodické linky sa musia odvíjať okolo root-tónu). Substituovať sa bude na interval od základného tónu v aktuálnom akorde, nie od toniky, na výsledku nesmie byť poznať že sa jedná o automatickú hudbu. Téma vyžaduje nielen vzdelanie v oblasti klasických šifier, ale hlavne hudobnej teórie. Pravidlá pre generovanie melódie budú závisieť práve od úrovne vzdelania v tejto oblasti. Vstupom by mala byť správa, výstupom MIDI (prípadne iný hudobný zápis). Vzhľadom na moju zahraničnú stáž, bude v prvom semestri prebiehať konzultácia cez internet. Vyžadovaná veľká miera samostatnosti. Úlohy: 1. Analyzovať techniky a metódy komponovania hudby 2. Vytvoriť substitučnú šifru (správa-melódia) 3. Vytvoriť nástroj na automatizované vytváranie komplexnejšej hudby zo správ (zohladňovať zmeny akordov pridať basovú linku atď.) 4. Otestovať riešenie na konkrétnej skladbe

Kľúčové slová: hudba, kryptosystém, automatizácia, komponovanie

ABSTRACT

SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA
FACULTY OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY

Study Programme:	Applied Informatics
Bachelor Thesis:	Composing music from message
Autor:	Pavol Sobota
Supervisor:	Ing. Peter Špaček
Consultant:	Ing. Eugen Antal, PhD.
Place and year of submission:	Bratislava 2020

The goal of this final thesis is to create a substitution cipher for encoding a message into music. Message has to be encoded into melody, not into the sound recording, this is not a steganography model. Music composition has to comply with the rules of pop-music (for example 3, 4 chords, pentatonic scales, melody depends on the root-note). The subject of the substitution will be an interval positioned around the base-note of current chord, not the tonic note. The outcome has to be indistinguishable from man-made music. The topic requires knowledge not only in the area of classical ciphers, but mainly in music theory. Melody generating rules will depend mostly on the knowledge in this area. Message should serve as an input and MIDI (or similar) file as an output. Tasks: 1. Analyse techniques and methods used for music composition. 2. Create a substitution cipher (message-melody) 3. Create a tool for automated composition of more complex music from messages (keep in mind, the changing of chords, add a base line etc.) 4. Test the solution on specific music piece.

Key words: music, cryptosystem, automatisation, composition

Vyhlásenie autora

Ja Pavol Sobota čestne vyhlasujem, že som bakalársku prácu Automatické komponovanie hudby podľa správy (šifrovanie do hudby) vypracoval na základe poznatkov získaných počas štúdia a informácií z dostupnej literatúry uvedenej v práci.

Vedúcim mojej bakalárskej práce bol Ing. Peter Špaček.

V Bratislave dňa 31.05.2020

.....

podpis autora

Poďakovanie

Veľká vďaka a uznanie patrí v prvom rade vedúcemu práce Ing. Petrovi Špačekovi, za jeho rozsiahlu kreativitu, odborné rady a obetovaný čas počas zahraničnej stáže a konzultácií. Rovnako sa chcem poďakovať Michalovi Malčovskému za pomoc pri komponovaní hudby a perspektívu umelca. Práca je venovaná informatikom nachádzajúcim porozumenie v hudbe.

Obsah

Zoznam obrázkov a tabuliek

Úvod

Kryptológia je veda, ktorá sa zaoberá metódami ako ochrániť správy pred potenciálnym protivníkom. Taktiež sa zaoberá štúdiom ako ochrániť použité metódy pred známymi spôsobmi ich prelomenia (1). Od najstarších známych spôsobov ako Cézarova šifra, alebo šifra Skytalé (1) sa postupom času vytvárali nové, komplexnejšie metódy utajenia informácie. V dnešnej dobe, je už každodenným štandardom používať utajenú komunikáciu prostredníctvom internetu.

V priebehu histórie sa objavovali rôzne spôsoby, ako pred protivníkom utajiť fakt, že tajná správa vôbec existuje. Často používaným spôsobom bolo napríklad písanie správy neviditeľným atramentom. Veda, ktorá sa zaoberá týmto spôsobom utajenia informácie sa nazýva steganografia (1). Idea ukrytia informácie v hudbe existuje už od obdobia renesancie. Jedna z najstarších zmienok, ktorá hovorí o využití hudby týmto spôsobom, patrí skladbe *Missa Hercules Dux Ferrariae* od Josquina des Preza z roku 1480 (2). Mnoho iných skladateľov a kryptológov vytvorilo množsvo rôznych spôsobov, pomocou ktorých sa v hudbe dá utajiť informácia. Najčastejším problémom, s ktorým sa pri tvorbe týchto metód stretli, bolo nedodržanie estetickej stránky, ktorú by mala obsahovať výsledná hudba.

Spojenie hudby a matematiky si uvedomoval už Pytagoras v 6. storočí p.n.l., kedy prišiel k záveru, že dva zvuky znejúce súčasne vyvolávajú estetický dojem, ak pomer ich frekvencií (vtedy dĺžok strún hudobného nástroja) je racionálne číslo. Využívaniu podobných vzťahov sa venuje algoritmické komponovanie hudby, ktoré bolo najviac rozvinuté v posledných dvoch storočiach (3). Cieľom algoritmického komponovania je najmä automatizácia hudobnej tvorby.

V tomto bode vstupuje do riešenia problematiky naša práca. V práci sa pokúsime navrhnúť hudobný kryptosystém, ktorý bude spájať prvky utajenia informácie v hudbe a automatickej hudobnej kompozície. Dôležitým faktorom pri tvorbe tohto kryptosystému bude dodržať estetickú kvalitu hudby. Hudba by mala byť vhodná na prenos širokou verejnosťou využívanými multimediálnymi kanálmi ako rádio, televízia, bez toho aby potenciálny protivník vydedukoval, že v nej môže byť ukrytá správa. Čitateľ tejto práce nám bude musieť odpustiť skutočnosť, že estetickú stránku hudby nie sme schopní exaktne definovať, ani dokázať zaužívanými metódami matematických dôkazov. Jej prítomnosť vieme dokázať len na základe experimentov, alebo prieskumu názorov širokej verejnosti.

1 Úvodné pojmy

Kryptosystém – V kryptológii označuje skupinu algoritmov potrebných na implementovanie určitej bezpečnostnej služby. Typicky pozostáva z algoritmov zabezpečujúcich generovanie kľúča, šifrovanie a dešifrovanie.
Logická relácia – Dôkazová metóda v sémantike programovacích jazykov, ktorá hovorí o tom, že dve denotačné sémantiky sú ekvivalentné.
Otvorený text (OT)- Text pred zašifrovaním v pôvodnom tvare (1).
Zašifrovaný text (ZT)– Text po zašifrovaní. Zmysel textu je osobe, ktorá šifru nepozná utajený (1).
Binárny reťazec – Sekvencia bitov. (0,1) Používa sa na reprezentovanie rôznych dátových typov ako text, obrázok, súbor atď.
Architektúra systému – Konceptuálny model, ktorý definuje štruktúru a správanie informatického systému. Obsahuje opis jednotlivých komponentov a podsystémov.
TSA – Štandardná telegrafná abeceda obsahujúca 26 znakov od „A“ po „Z“ (1).
Suffix – Jazyková morféma, ktorá sa pripája za koreň slova. (prípona)
Offset – Hodnota relatívnej vzdialenosti medzi dvoma entitami.

2 Úvod do hudobných kryptogramov

Existujú dva rozdielne prístupy riešenia problematiky ako ukryť informáciu do hudby.

Jeden zo spôsobov využíva určité obmedzenie v ľudskom sluchovom systéme. Ľudské ucho totižto nie je schopné odlíšiť od seba rozdielne zvuky, ak sa nachádzajú v istom úzkom intervale. Princíp tejto metódy spočíva v tom, že utajené bity informácie sú prenesené pomocou zvukového signálu, ktorý zaznieva práve v intervaloch, v ktorých ich človek nedokáže odlíšiť od zvukov patriacim hudbe. Táto metóda sa nazýva **psychoakustické maskovanie** a patrí do kategórie **steganografie.** Osoba prijímajúca takto utajenú správu ju vie pomocou digitálnych zariadení odpočúvať a zo signálov spätne extrahovať. Tento systém sa širokospektrálne využíva v praxi. Našim cieľom je ale vytvoriť inovatívne, odlišné riešenie pre túto problematiku (4).

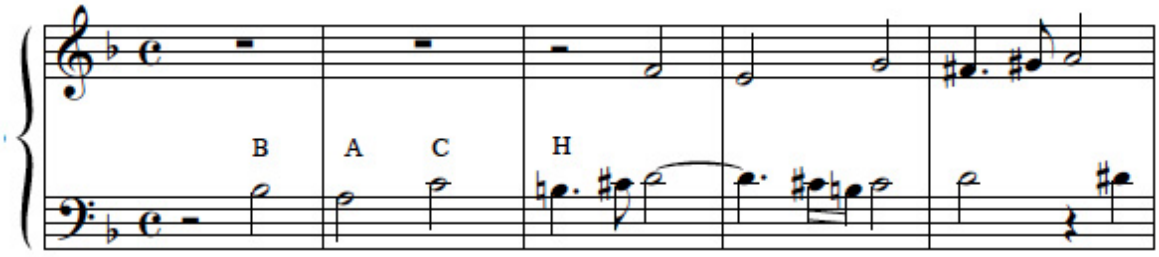
Druhým zo spôsobov je použitie **hudobného kryptogramu** (5)**.** Hudobný kryptogram je sekvencia hudobných nôt, ktoré sú v určitej logickej relácii s vonkajšou nehudobnou informáciou, obvykle textom. Zásadným rozdielom medzi hudobným kryptogramom a steganografickým modelom je to, že v steganografickom je prenášaná tajná informácia ukrytá pred detekciou, ale štruktúra dát nie je pozmenená. V hudnom kryptograme je štruktúra dát zmenená na samotnú hudbu. Hudba je pre potenciálneho útočníka voľne dostupná a viditeľná, ale jej význam je skrytý. Našou prácou bude teda vytvoriť unikátny hudobný kryptogram, schopný skryť tajnú informáciu pred detekciou.

Aby sme o tvorbe hudobných kryptogramov získali viac poznatkov, je potrebné spomenúť aké hudobné kryptogramy boli navrhnuté v minulosti. Týmito kryptogramami sa pri riešení našej problematiky môžeme inšpirovať, alebo sa vyhnúť chybám a nedostatkom, ktoré sa v nich vyskytli.

2.1 Podpis v motíve

2.1.1 Johann Sebastian Bach

Najjednoduchším a jedným z prvých spôsobov ukrytia informácie v hudbe bolo podpísanie sa autora v motíve skladby. Autor pridal do melódie skladby skupinu nôt, ktoré sa vo vtedajšom pomenovaní tónov zhodovali so znakmi tvoriacimi jeho meno. Medzi najznámejšie skladby s podpisom v motíve patrí *Fuge XV* od J. S. Bacha. Jeho meno B-A-C-H je zakomponované v melódii ako môžeme vidieť na obrázku 1 (5).



Obr. 1 Výňatok zo skladby Fúga XV od J. S. Bacha

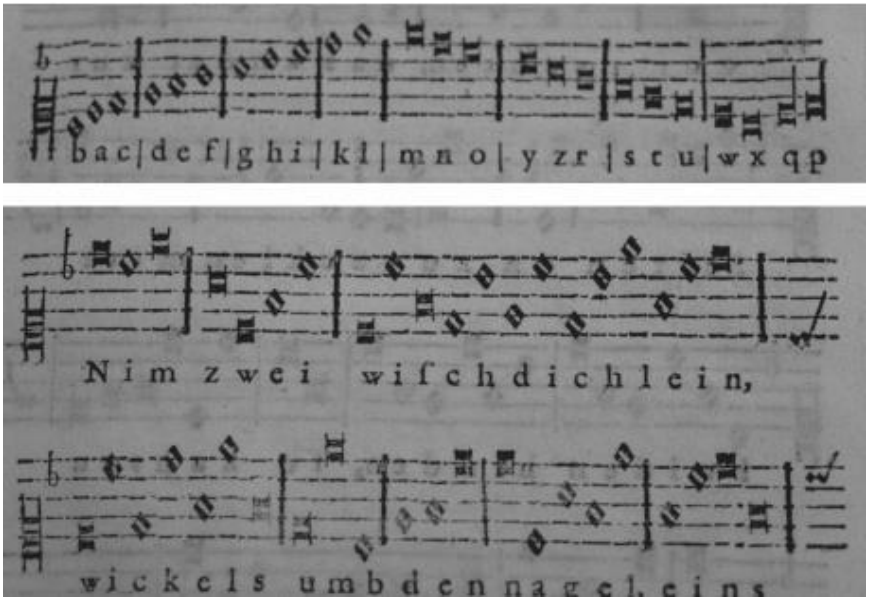
2.1.2 Johannes Brahms

Ďalší zaujímavý odkaz v motíve skladby patrí J. Brahmsovi, ktorý v diele Fuga pre organ v As mole použil meno spisovateľky Gisely von Arnim. Znaky z jej mena pretransformoval na tóny Gis, E, La. Solmizačná slabika La je zhodná s tónom A (5).

2.2 Substitučná šifra

2.2.1 Gaspar Schott

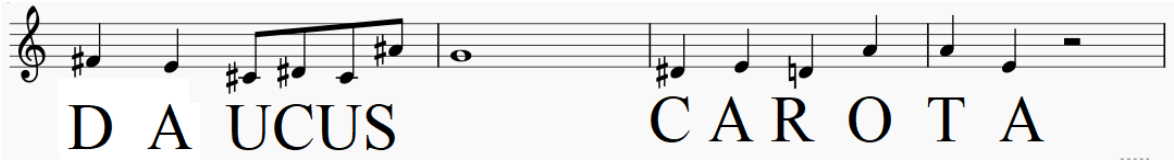
Sofistikovanejší spôsob ukrytia informácie v hudbe opísal Gaspar Schott vo svojom diele *Schola Steganographica* (6). Vytvoril substitučnú šifru (1), kde je každému písmenu abecedy priradená jedna nota. Za každý znak v jeho šiforvanej správe jednoducho v skladbe napíše notu prislúchajúcu tomuto znaku. Táto metóda ukrývania informácie v hudbe nie je prispôsobená na to, aby výsledná hudba znela esteticky. Jej cieľom je len prenos utajenej správy prostredníctvom zašifrovania do hudobného zápisu. Originál šifry je možné vidieť na obrázku 2.



Obr. 2: Ukážka šifry, ktorú navrhol G. Schott

2.2.2 Hermetic Order of the Golden Dawn

Podobný spôsob tvorby hudobného kryptogramu opisuje Ezra Sandzer-Bell v knihe *Astromusik* (7). Predmetom jeho výskumu bola šifra, ktorú používal v prvej polovici dvadsiateho storočia americký kult Hermetic Order of the Golden Dawn. Člen tohoto kultu vytvoril šifru, kde každá nota v celom 12 notovom hudobnom systéme reprezentovala buď farbu, planétu, znamenie zverokruhu alebo znak hebrejskej abecedy. Sandzer-Bell opisuje ako je tento spôsob použiteľný na tvorenie piesní. Každé slovo je možné preložiť do hebrejčiny a následne použiť tento kryptogram na vytvorenie hudby.



Obr. 3: Príklad použitia šifry kultu Hermetic Order of the Golden Dawn

2.3 Morseov kód

V roku 2010 kolumbijský tvorca reklám Juan Carlos Ortiz navrhol hudobný kryptogram, pomocou ktorého bolo možné kontaktovať zajatých vojakov kolumbijskej vlády. Princíp tohoto kryptogramu spočíval v tom, že v refréne vytvorenej piesne bola pomocou opakovaného hrania jedného tónu zašifrovaná správa formou Morseovho kódu. Dôležitým faktom v použití tejto metódy bol predpoklad, že rebelujúca ofenzíva je tvorená najmä roľníkmi a osobami z chudobných pomerov, a tým pádom Morseov kód nepozná. V práci od J. Maysha (8) je opísaný postup tvorenia tohoto kryptogramu. Jeden z dôležitých problémov, s ktorými sa Ortiz stretol bolo určenie koľko Morseových slov je možné zašifrovať do refrénu, aby pieseň nestratila svoj estetický charakter a dostala sa medzi populárne piesne hrané v rádiu. Znalec Morseovho kódu dokáže za minútu prečítať zakódovaných 40 slov. Toto bola ale príliš veľká rýchlosť hrania tónov pre populárnu hudbu a tak zhodnotil, že maximum zakódovaných slov bude 20. Správa nachádzajúca sa v piesni teda znela “19 LIBERADOS. SIGUEN USTEDES. ANIMO.” a jej znenie bolo trikrát zopakované. V preklade znamenala “19 zachránených. Ste na rade. Povzbuďte sa.”

3 Hudobná teória

Aby čitateľ vedel podrobne pochopiť funkcionalitu systému, musí porozumieť základným hudobným pojmom, s ktorými budeme pracovať. Jednotlivé algoritmy zodpovedné za komponovanie hudby budú reprezentovať tieto pojmy pomocou množín a ich prvkov. Vzťahy, ktoré sú medzi nimi je ale potrebné uviesť pred návrhom systému.

3.1 Základné hudobné pojmy

Tón reprezentuje jednoduché, pravidelné chvenie zdroja zvuku. Od zvukového šumu sa líši tým, že tvorí obzvlášť ušľachtilý zvukový vnem. Z akustického hľadiska má tón štyri vlastnosti: silu, farbu, výšku a dĺžku. V práci sa budeme zaoberať najmä výškou a dĺžkou (9).

Výška tónov vyjadruje, koľko kmitov zvuk vykoná za určitú časovú jednotku. Ľudské ucho je schopné počuť zvuky v rozmedzí 16 Hz až 20 kHz. Štandardná klasická hudba používa dokopy 88 výsekov tónov, ktorých rozdiely sú tak zrejmé, že ich je človek schopný rozoznať. Výšková vzdialenosť medzi jednotlivými tónmi sa nazýva **interval**. Interval medzi tónom s určitou výškou a druhým s dvojnásobnou výškou toho prvého nazývame **oktáva**. Tento interval je rozdelený na dvanásť tónov. Tieto opakujúce sa tóny sú označené písmenami abecedy: a, a#, b (h), c, c#, d, d#, e, f, f#, g, g#. Aby sme určili, ktorý tón sa nachádza v ktorej oktáve používame zápis C1, B2 atď. Najmenšia možná vzdialenosť medzi výškami jednotlivých tónov sa nazýva poltónová vzdialenosť (9).

Stúpajúci alebo klesajúci usporiadaný rad tónov od základného tónu, po jeho oktávu nazývame **stupnica.** Všetky tóny obsiahnuté v skladbe, ktoré prislúchajú určitej stupnici nazývame **tónina. Durové** tóniny sú zostavené z ôsmich tónov. Medzi tretím a štvrtým, siedmym a ôsmym tónom je poltónová vzdialenosť. Medzi ostatnými tónmi je vzdialenosť dvojnásobná, teda celotónová. Jednotlivé tóny v stupnici majú rôzny hudobný význam, preto dostali aj svoje konkrétne označenia (10).

I	Tonika
II	Supertonika
III	Medianta/Modal
IV	Subdominanta
V	Dominanta
VI	Superdominanta
VII	Subtonika
VIII	Tonika

Obr. 4: Názov stupňa tónu v stupnici

Dĺžka tónu predstavuje čas počas ktorého nechávame zaznievať tón. Hodnoty dĺžky nôt sa ustálili na dnes používané celé, polové, štvrťové, osminové a šestnástinové noty. Každá ďalšia hodnota je vždy polovica časového priebehu tej predošlej. Tieto časové pomery porovnávajú dĺžky tónov len relatívne v závislosti od tempa skladby. **Bodkovaná nota** je nota, ktorá má dĺžku jeden a pol krát dlhšiu ako jej príslušná nota bez bodky. Napríklad ak štvrťová nota bude hraná jednu sekundu, bodkovaná štvrťová nota jeden a pol sekundy. **Pomlčka** označuje miesto v skladbe, kde sa momentálne nehrá žiaden tón. Rovnako ako tóny aj pomlčky majú svoje dĺžky (9).

3.2 Rytmus

Takt je základnou rytmickou a metrickou jednotkou skladby. Takt je vnútorne členený na úseky rovnakej dĺžky, teda **doby**. Pri dobe vždy udávame, tónom akej dĺžky môže byť reprezentovaná. Teda ak označíme dobu ako štvrťová, môže byť reprezentovaná jedným štvrťovým tónom alebo dvoma osminovými tónmi atď. Na základe toho na koľko dôb takt rozdelíme, je potom takt aj označený. Napríklad ak je rozdelený na dve doby s dĺžkou štvrťovej noty, hovoríme o dvoj-štvrťovom takte (10).

Rytmus je rozdelenie rytmických hodnôt, ktoré vnútorne vypĺňajú takty. V rámci jedného taktu môže byť usporiadanie rytmických štruktúr rôzne. Hlavné však je, že ich výsledná hodnota trvania musí byť vždy presne rovnaká. Zákonitosti platiace v rytme sú **metrum** a **rytmická pulzácia**. Metrum je pravidelné striedanie prízvučných a neprízvučných dôb, do štruktúry taktu, vyplneného konkrétnym rytmom vnáša poriadok. Rytmická pulzácia je sled pravidelne sa opakujúcich, rovnakých úsekov, z pohľadu porovnania dĺžky tónov (9).

Tempo skladby znamená základnú prednesovú rýchlosť hudby. Pod touto rýchlosťou si predstavujeme striedanie jednotlivých dôb za určitú časovú jednotku, obvykle minútu. Tempo úzko súvisí s témou skladby, veselšie skladby sú hrané rýchlejšie. Najpresnejšie sa určuje metronomicky. To obvykle podľa toho koľko štvrťových nôt vieme zahrať po sebe za dobu jednej minúty. Ak tempo určíme na 120 úderov za minútu, štvrťová nota bude mať dĺžku pol sekundy (9).

3.3 Melódia

Melódia je jeden z najzákladnejších elementov v hudbe. Je tvorená reťazcom po sebe nasledujúcich tónov. Tento reťazec nie je akýkoľvek. Musí byť sformovaný tak, aby melódia znela príjemne a rytmicky správne. Okrem hlavných tónov melódie sa v skladbe môžu nachádzať iné tóny, ktoré slúžia ako skrášlenia a ozdoby, aby skladba znela viac komplexnejšie (11).

Melodický pohyb nám hovorí o ďalších užitočných vlastnostiach melódie, ktoré používame v práci. Konkrétne o tom, ako rýchlo melódia klesá alebo stúpa čo sa týka výšky jej tónov. Melódia, ktorá stúpa alebo klesá v najmenších intervaloch tónových alebo poltónových medzi dvoma tónmi sa nazýva **konjunktná.** Prechod z jedného tónu na druhý v konjunktnej melódii budeme nazývať podľa anglického výrazu **"step"**. Melódia, ktorá stúpa vo väčších intervaloch medzi jednotlivými tónmi sa nazýva **disjunktná.** Prechod medzi dvoma tónmi budeme nazývať **"leap"**. Všeobecne je melódia tvorená kombináciou konjunktných a disjunktných melodických pohybov (11).

Ako melodickú **frázu** označujeme zoskupenie tónov v melódii. Fráza je ukončená melodicko-harmonickou konfiguráciou tónov nazývanou **kadencia**. Vyvoláva dojem ukončenia, alebo očakávania na pokračovanie s iným motívom. V bežnej populárnej hudbe sú frázy dlhé väčšinou 4 takty. Tak to bude aj v našej práci. Dve frázy hrané po sebe budeme nazývať **perióda.**

3.4 Harmónia

Súzvuk je skupina aspoň dvoch súbežne znejúcich tónov. Súzvuky skladajúce sa aspoň z troch tónov sa nazývajú **akordy**. Tóny tvoriace tento akord sa patrične nazývajú potom **akordické tóny.** Kvintakord je najjednoduchší základný akord. V našej práci budeme používať najmä kvintakordy. Jeho najnižší tón označujeme ako základný alebo basový tón. Podľa neho aj nazývame daný akord. Ďalšie 2 tóny sú tvorené z tónov, ktoré sú od basového tónu vzdialené prvý o 2 tóny smerom vyššie a druhý o 4 tóny. **Progresia akordov** je pojem, podľa ktorého určujeme v akom poradí sa budú v našej skladbe akordy vyskytovať (12).

Harmóniu si v skratke môžeme predstaviť ako jav v hudbe, kedy rôzne nástroje hrajú rôzne tóny, ale ich spoločný zvuk tvorí určitý akord. Pre príklad klavír bude hrať tón C, husle E a gitara G. Spolu tvoria kvintakord C alebo v tónine C dur akord I. stupňa.

Basová linka označuje v hudbe časť melodickej linky, je hraná na spodku zvukového rozhrania skladby. Obvykle je to interval nižší o jeden a pol oktávy ako tón C5. Úlohou basovej linky je zdôrazňovať harmóniu a harmonický pohyb. Aby bola basová linka v súlade s melodickou linkou hranou vo vyššej polohe, musia ich tóny na určitých miestach tvoriť spoločný akord. V praxi sa to dá najjednoduchšie docieliť tým, že v melodickej linke bude v prvej a tretej dobe v každom takte zahratý jeden z akordických tónov akordu, v ktorom sa práve nachádzame.

3.5 Štruktúra krátkej piesne

Počas histórie sa v hudbe vyvinulo množstvo hudobných žánrov. Diela a skladby patriace do týchto žánrov boli skomponované v rôznych formách a štruktúrach. Typická štruktúra jednoduchej novodobej piesne pozostáva zo sekcii intro, outro, verš, refrén a bridge (13). Sú v rozložení I-V-R-V-R-B-R-O. Často sa vyskytujúca dĺžka jednej sekcie je 32 dôb. Viac sa môžete dočítať v tomto zdroji (14).

Každá sekcia je význačná svojim motívom alebo tematikou, ktorú vyjadruje. **Intro,** úlohou tejto sekcie je zachytiť pozornosť poslucháča. Taktiež ustanoviť tempo, nastoliť rytmus a všeobecný motív piesne. Intro býva obvykle pomalšie ako zvyšok piesne. **Verš** je priestor pre rozvoj tematiky piesne, lyricky autor opisuje detaily piesne, príbeh, udalosti emócie atď. Hudobne je často tvorený akordami, ktoré obsahujú niektoré z nôt hlavného kvintakordu toniky. **Refrén** vytvára melodický, rytmický alebo harmonický kontrast k veršu. Znenie motívu sa v refréne obvykle viackrát zopakuje, alebo je naň kladený obzvlášť výrazný dôraz. Refrén má z pravidla väčšiu hudobnú intenzitu ako verš. Úlohou **bridge-u** je prelomiť repetitívny vzor piesne a udržať si pozornosť poslucháča. V mnohých prípadoch je to pomocou vytvorenia kontrastu k predošlej hranej sekcii. **Outro** je spôsob ukončenia piesne. Nachádzajú sa v ňom prvky ako postupné spomaľovanie tempa, stišovanie hlasitosti hudby alebo uvoľnenie napätia s prvkom kadencie.

4 Technické špecifikácie

4.1 Java

Na vytvorenie systému sme použili programovací jazyk Java. Je to na základe skúseností autora s týmto jazykom a knižnice JFugue, ktorá vie širokospektrálne pracovať s hudbou.

4.2 MIDI štandard

Existuje množstvo spôsobov prenosu a ukladania digitálnej audio informácie a rôzne súborové formáty na to prispôsobené. Pre našu prácu sme sa rozhodli použiť formát .mid, ktorý je nosičom dát pre **MIDI** komunikačný protokol. MIDI protokol bol vytvorený priamo na reprezentáciu hudby. Na rozdiel od väčšiny audio formátov, ktoré reprezentujú zvuk pomocou prevodu frekvencie, hlasitosti a pod. na digitálnu informáciu, MIDI je tvorený dátami, ktoré špecifikujú rôzne hudobné inštrukcie. Medzi inštrukcie patrí dĺžka, výška noty, sila, vibrato a informácia o tempe skladby. Okrem týchto základných inštrukcií špecifikuje rôzne iné funckie, ktoré pomáhajú reprezentovať hudbu. Mnohé z nich v práci využívame. MIDI ponúka 16 samostatných kanálov. Každý z nich nesie vlastný set hudobných inštrukcií. MIDI taktiež ponúka 128 rôznych hudobných nástrojov, ktoré môžu byť pridelené danému kanálu. Pre lepšie pochopenie si môžeme MIDI predstaviť ako sériu nôt, ktoré majú byť zahrané hudobným nástrojom.

4.3 Knižnica JFugue

Pre jazyk Java je vytvorených viacero knižníc, ktoré pracujú s hudbou. Knižnica JFugue nám prišla ako dostatočne všestranná na to, aby sme pomocou nej vytvorili hudbu dostatočnej kvality. Taktiež v JFugue sú jednotlivé kroky hudobnej kompozície priamo prispôsobené na prácu s MIDI (15).

4.3.1 MusicString

V knižnici JFugue je sled tónov reprezentovaný pomocou objektu MusicString. MusicString si môžeme predstaviť ako premennú typu String, ktorá je potom parsovaná na podreťazce reprezentujúce jednotlivé tóny, ktoré môžu byť prehraté, alebo zakódované do výstupu MIDI. Jednotlivé tóny sú oddelené medzerou. Napríklad, ak chceme zakódovať tón A v oktáve 5 s dĺžkou jedna polovica doby tak napíšeme:

```
Pattern pattern = new Pattern("G5h");
```

alebo číselným zápisom:

```
Note note = new Note(67, 0.5);
```

Číselný zápis použijeme pri vytváraní rôznych algebraických štruktúr, kde vzťahy medzi jednotlivými notami môžeme reprezentovať pomocou matematických operácií. Toto je tabuľka všetkých tónov, ktoré je možné zakódovať do MusicString.

Octave	C	C#/D*b*	D	D#/E*b*	E	F	F#/G*b*	G	G#/A*b*	A	A#/B*b*	B
0	0	1	2	3	4	5	6	7	8	9	10	11
1	12	13	14	15	16	17	18	19	20	21	22	23
2	24	25	26	27	28	29	30	31	32	33	34	35
3	36	37	38	39	40	41	42	43	44	45	46	47
4	48	49	50	51	52	53	54	55	56	57	58	59
5	60	61	62	63	64	65	66	67	68	69	70	70
6	72	73	74	75	76	77	78	79	80	81	82	83
7	84	85	86	87	88	89	90	91	92	93	94	95
8	96	97	98	99	100	101	102	103	104	105	106	107
9	108	109	110	111	112	113	114	115	116	117	118	119
10	120	121	122	123	124	125	126	127				

Obr. 5: Tabuľka numerických hodnôt tónov

Pomlčky môžeme reprezentovať zápisom znaku **R** (rest), za ktorým nasleduje znak vyjadrujúci hodnotu dĺžky tejto pomlčky, všetky znaky vyjadrujúce dĺžku je možné vidieť na obrázku 6.

Duration	Character
whole	w
half	h
quarter	q
eighth	i
sixteenth	s
thirty-second	t
si**x**ty-fourth	x
one-twenty-eighth	o

Obr. 6: Tabuľka označení pre dĺžky tónov

4.3.2 Pattern

Ako sme si mohli všimnúť vyššie, na uloženie MusicStringu sme použili objekt Pattern. Funkciou Patternu je možné obsiahnuť sled tónov a uložiť ich v pamäti ako objekt. K tónom potom môžeme pomocou jeho metód pristupovať, meniť ich alebo mazať. Jednotlivé patterny sme schopní spájať a tým vytvárať ucelenú skladbu.

```
Pattern pattern0 = new Pattern("10q 15h 11q 14h 10q 15h");
Pattern pattern1 = new Pattern("16h");

Pattern song1 = new Pattern();
```

```
song1.add(pattern0,pattern1);
```

4.3.3 Práca s MIDI

JFugue ponúka podporu súborov MIDI nasledovným spôsobom. Knižničné metódy dané MIDI uložia, parsujú a vložia do objektu Pattern.

```
MidiFileManager.loadPatternFromMidi(new File("music-file.mid
"));
```

JFugue môže MIDI súbory aj vytvárať a to pomocou príkazu.

```
MidiFileManager.savePatternToMidi(pattern, new File("music-
file.mid "));
```

5 Špecifikácie pre návrh

V nasledujúcich kapitolách uvedieme náš prístup k riešeniu problematiky z praktického hľadiska. Najprv aké pravidlá sme si určili, ako sme postupovali pri návrhu a na koniec vysvetlíme konkrétne algoritmy, ktoré sme navrhli.

V tejto kapitole spomenieme aké požiadavky a obmedzenia sme zvolili pre kryptosystém. Počas vývoja architektúry systému sa podľa potreby formulácia niektorých požiadaviek menila. V tejto časti sú spomenuté prvotné a najviac podstatné požiadavky. Na základe toho, že predmetom našej práce je hudobný kryptosystém, sme požiadavky rozdelili na kryptografické a hudobné.

Kryptografické. Šifra má na vstupe text alebo binárne dáta. Spĺňa Kerckhoffov princíp (16) a je odolná voči praktikám používaným na lúštenie klasických šifier. Šifra je zostavená tak, že po rozšifrovaní sme schopní úspešne získať rovnakú informáciu, ktorú sme zašifrovali.

Hudobné požiadavky nám vytvárajú usmernenie, ako hudbu komponovať, na ktoré oblasti sa v procese zamerať, aby bol hudobný výstup čo najviac príjemný na počúvanie, poprípade nerozoznateľný od bežnej, širokou verejnosťou počúvanej hudby. V zmysle elementov a aspektov, z ktorých sa hudba skladá, sme ďalej hudobné požiadavky rozdelili na melodické, harmonické a rytmické.

Melodické. Melódia hraná počas celej skladby bude v jednej rovnakej tónine. Tónina bude buď zvolená alebo vygenerovaná na základe vstupu. Tóniny, v ktorých sa budú pohybovať tóny melódie budú durové alebo molové (9). To z dôvodu jednoduchej práce s nimi a širokom využití v praxi. V melódii by sa mali vyskytovať prvky kadencie a opakovanie určitých melodických sekcií. Melodický pohyb by nemal obsahovať po sebe nasledujúcich veľa disjunktných nôt, ktoré by v skladbe vyvolali pocit stresovosti. Rozsah tónov v melódii by nemal zasahovať do príliš vysokých, alebo príliš nízkych tónov. Z dôvodu, že komponovanie v týchto intervaloch nie je jednoduché a taktiež, aby melódia mohla byť potenciálne zaspievaná.

Harmonické. Skladba bude dodržiavať vopred určenú progresiu akordov. Taktiež by sme mali skomponovať basovú linku, ktorá bude pre harmóniu oporným prvkom. Basová linka bude súčasne vyjadrovať, aký akord z progresie akordov momentálne hráme. Melódia by mala byť s harmóniou v akordickom súlade aspoň v najjednoduchšom prevedení. Teda v melódii v prvej a tretej dobe každého taktu bude prvý tón akordický.

Rytmické. Jednotlivé takty v periódach budú rozdelené na rovnaký počet dôb. Vo všetkých dobách bude súčet dĺžok tónov v dobe obsiahnutých rovnaký. V skladbe sa bude nachádzať určitý typ opakovania rytmických štruktúr.

6 Inkrementálny postup navrhovania

Pri návrhu systému sme pracovali inkrementálne a v každej fáze sme systém vylepšovali. Sofistikovanosť riešenia problematiky závisela od znalostí autora v oblasti klasických šifier, ktoré pri návrhu postupne získaval. Taktiež aj od schopnosti riešiť problémy, ktoré sa vyskytli v predošlých verziách. Pre pochopenie výsledku je preto treba sledovať myšlienkový pochod autora. V tejto kapitole spomenieme podstatnejšie časti vývoja.

6.1 Jednoduchá substitúcia

Na začiatku sme použili podobný spôsob šifrovania, ako viacerí hudobní skladatelia z histórie. V našej architektúre mala šifra formu **jednoduchej substitúcie** (1), kde jeden znak otvoreného textu bol reprezentovaný ako tón v melódii s konkrétnou výškou, nami zvolenou na základe experimentov. V tomto štádiu bol vstupom do šifry len krátky text v TSA. V snahe dať náhodnému hudobnému výstupu nejakú formu, boli v samohlásky šifrované ako tóny s vyššou výškou a spoluhlásky s nižšou. Hudobný výstup mal aj napriek tomu dojem náhodnosti a zvolenú šifru bolo možné prelomiť využitím **frekvenčnej analýzy** (1).

6.2 Rozdelenie na dva moduly

Za účelom zvýšenia bezpečnosti sme otvorený text, na základe ktorého sa generuje hudobný výstup, potrebovali spraviť odolným voči frekvenčnej analýze. Túto vlastnosť vieme získať pomocou šifrovania otvoreného textu do formy reťazca znakov, ktorý je svojou formou nerozlíšiteľný od náhodného. Ak na to použijeme správne zvolenú šifru, systém bude spĺňať aj **princíp konfúzie a difúzie** (17) . Rozhodli sme sa teda, že vstupný text bude šifrovaný v závislosti od unikátneho tajného kľúča a tým pádom bude spĺňať **Kerckhoffov princíp** (16).

Na základe tejto zmeny sme architektúru systému rozdelili na dva separátne moduly. Jeden bude zabezpečovať spomenuté šifrovacie úkony. Druhý je navrhnutý tak, aby z reťazca, ktorý sa javí ako náhodný skomponoval hudobný výstup. Rovnako dešifrovanie je vykonané pomocou reverzných úkonov. Teda najprv je hudba dekomponovaná na reťazec, ktorý sa javí ako náhodný a potom je reťazec dešifrovaný pomocou kľúča na otvorený text.
V tomto štádiu je potrebné si tieto moduly a ich procesy formálne zadefinovať.

Majme množiny: otvorený text $\mathbf{T_1} = \{x_1, x_2 \dots x_n\}$, zašifrovaný text $\mathbf{T_2} = \{x_1, x_2 \dots x_n\}$, kľúč **k**, ktorý bude pre jednoduchosť zapamätania tvorený znakmi využívanými svetovými jazykmi: {“A”, ”B” … ”Z”, ”a”, “b” … “z”, “0” … “9”}.

Potom:

(1) $\mathbf{s_1}$ je zobrazenie $\mathbf{s_1(k) : T_1 \rightarrow T_2}$. Modul šifrovania vykonávajúci proces $\mathbf{s_1}$ označme $\mathbf{S_1}$.

(2) $\mathbf{s_2}$ je **inverzné** zobrazenie k $\mathbf{s_1}$, $\mathbf{s_2(k) : T_2 \rightarrow T_1}$. Modul dešifrovania vykonávajúci proces $\mathbf{s_2}$ označme $\mathbf{S_2}$.

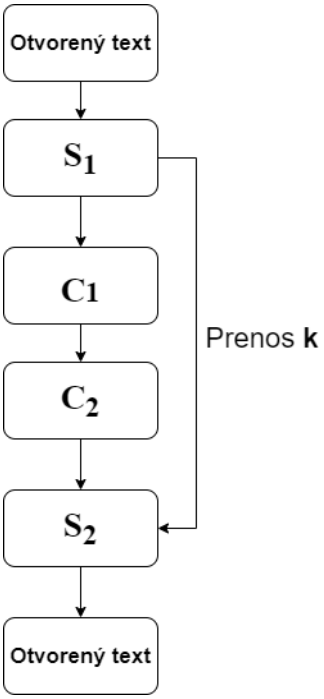
Majme množinu: $\mathbf{H} = \{h_1, h_2 \dots h_n\}$ tvorenú prvkami, ktoré reprezentujú určitý hudobný pojem, notu, označenie, dej atď.

Potom:

(1) $\mathbf{c_1}$ je **injektívne** zobrazenie $\mathbf{c_1 : T_2 \rightarrow H}$. Modul komponovania hudby vykonávajúci proces $\mathbf{c_1}$ označme $\mathbf{C_1}$.

(2) $\mathbf{c_2}$ je **surjektívne** a **inverzné** zobrazenie k $\mathbf{c_1}$, $\mathbf{c_2 : H \rightarrow T_2}$. Modul dekomponovania hudby vykonávajúci proces $\mathbf{c_2}$ označme $\mathbf{C_2}$.

Konkrétne prvky tvoriace množiny definujeme v nadchádzajúcich kapitolách.
Architektúru tvorenú týmito modulmi môžeme vidieť na obrázku 7.



Obr. 7: Schéma architektúry systému

V tejto verzii modul **C1** komponoval hudbu podobným spôsobom ako metóda nazývaná **expanzná substitúcia** (1). Teda znak na vstupe bol zakódovaný do výstupnej melódie ako jeden tón, viacero tónov rôznej dĺžky alebo zmena v melodickom pohybe. Tento vzťah bol reprezentovaný tabuľkou, kde znak zo vstupu bol indexom a hudobná štruktúra hodnotou pre daný index.

Aj keď sa náš systém touto metódou stal bezpečným, generovanie hudby bolo stále nedostačujúce. Tento problém sa nám podarilo vyriešiť až v ďalšej iterácii.

6.3 Vyšší level abstrakcie

Podstatnou myšlienkou v tejto verzii bolo uvedomiť si, že ak náš modul komponovania dokáže vytvoriť hudbu z náhodného reťazca znakov, je možné tento reťazec reprezentovať ako **binárny reťazec.** Táto reprezentácia nám ponúka hneď dve signifikantné pozitíva.

Prvé spočíva v tom, že vstupom do našej architektúry môže byť akákoľvek informácia v **bitoch**. Doteraz bol vstupom do systému len reťazec znakov abecedy. Teraz to môže byť textová informácia, obrázok, súbor atď. Architektúra systému musí byť aj patrične prispôsobená na prácu s binárnym reťazcom. V šifrovacom a dešifrovacom module treba správne zvoliť šifru, ktorej vstupom aj výstupom je bitová informácia.

Druhé pozitívum spočíva v tom, že ak vstupom v module $\mathbf{C_1}$ je binárny reťazec, tak úkony, ktoré sa vykonávajú pri samotnom komponovaní môžu byť navrhnuté ako séria **rozhodovacích problémov,** alebo iných postupov, ktoré rozhodnú o tom aký výstup vygenerujú na základe bitov prevedených na číslo, reprezentujúce nejakú konkrétnu možnosť. Tieto metódy je možné potom reprezentovať pomocou konečných stavových automatov.

Na základe tejto architektúry môžeme dospieť k záveru, že výsledný hudobný výstup vytvárame pomocou transformácie **bitového vstupu** na určitú unikátnu **hudobnú štruktúru** vo výstupe. Tu sa nám vynárajú dôležité otázky, na ktoré si potrebujeme odpovedať a úsudok z nich aplikovať do architektúry a poprípade pridať do špecifikácie požiadaviek.

1. Ktoré konkrétne kroky komponovania hudby si zvoliť, aby ich hudobný výstup bolo možné získať len na základe operácií s bitovým vstupom, patrične zvoleným pre daný krok komponovania.
   Napríklad: Vieme vygenerovať konkrétne melodické jednotky, v konkrétnej dobe, s konkrétnou dĺžkou len na základe série bitov?

2. Zvoliť si také kroky, aby spätne pri dekomponovaní hudby bolo možné z danej hudobnej štruktúry získať presne ten istý bitový vstup, na základe ktorého sme štruktúru skomponovali.

3. Keďže pieseň má konečnú dĺžku a konečný počet jednotlivých nôt, akým spôsobom transformovať binárny reťazec, aby sme do hudby vedeli týmto spôsobom zakódovať čo najväčší počet bitov s tým, že hudba bude dodržiavať vopred určené pravidlá.

V tomto bode môžeme zadefinovať prvky tvoriace množiny $\mathbf{T_1}$ a $\mathbf{T_2}$ ako {0,1}^x, kde x je počet bitov, ktoré vieme kryptosystémom spracovať. Ten sa počas vývoja menil, avšak v jednej verzii, respektíve pre jednu skladbu musí ostať fixný.

Návrh modulu $\mathbf{C_1}$, do ktorého sme aplikovali tieto pravidlá, bol schopný generovať hudbu vyššej estetickej kvality, ako predošlé verzie. Hudba ale napriek tomu nespĺňala **rytmické** a **harmonické** požiadavky. Týmto sa dostávame k finálnej verzii, kde sme tieto požiadavky zakomponovali.

7 Kompletný návrh kryptosystému

7.1 Base32

Náš systému vie zašifrovať akúkoľvek informáciu reprezentovanú ako binárný reťazec. V tejto kapitole sa zameriame primárne na šifrovanie textovej informácie. Predpokladáme, že náš systém bude najviac užitočný práve na prenos správ. Aby sme vedeli preniesť čo najväčšie množstvo textovej informácie tvorenej predovšetkým symbolmi, potrebujeme zvoliť efektívny spôsob ako tieto symboly pretransformovať na binárny reťazec. V informatike existuje mnoho formálnych štandardov, ktoré definujú kódovaciu schému schopnú reprezentovať znaky používané v písaných jazykoch. Jedným z týchto štandardov je **Base32**. Toto kódovanie je schopné kódovať 26 znakov abecedy A-Z a číslice 2-7.

Value	Symbol	Value	Symbol	Value	Symbol	Value	Symbol
0	A	8	I	16	Q	24	Y
1	B	9	J	17	R	25	Z
2	C	10	K	18	S	26	2
3	D	11	L	19	T	27	3
4	E	12	M	20	U	28	4
5	F	13	N	21	V	29	5
6	G	14	O	22	W	30	6
7	H	15	P	23	X	31	7

Obr. 8: Base32 Abeceda

Base32 sme zvolili z toho dôvodu, že obsahuje všetky znaky TSA, pomocou ktorých je vo väčšine používaných jazykoch možné vyjadriť informáciu. Taktiež decimálnu hodnotu reprezentujúcu jednotlivý znak môžeme previesť na binárnu. Jej veľkosť bude 5 bitov, čo je postačujúca veľkosť na reprezentovanie jedného symbolu v našej architektúre.

7.2 AES 128

Na transformáciu dátovej informácie z **T_1** na **T_2** v module **S_1** a spätne v module **S_2** sme sa rozhodli použiť šifru **AES**. Patrí medzi blokové šifry. V práci používame štandard s veľkosťou bloku 128 bitov. Pre jednoduchosť sme zvolili mód Electronic Codebook **(ECB)** určený na šifrovanie otvoreného textu na základe kľúča. Ako kľúč používame reťazec s veľkosťou 128 bitov. Šifrovanie je prevedené bez paddingu, aby sa zachovala veľkosť blokov, a nevytváralo nám stratu v počte bitov, ktoré potom transformujeme do hudby.

AES používame najmä preto, že patrí medzi súčasné štandardy v kryptografii jej prelomenie je časovo náročné. Ďalšími dôvodmi sú princípy, ktoré AES zabezpečuje. Princíp konfúzie, ktorý zabezpečuje to, že vzťah medzi OT a ZT je veľmi komplikovaný, teda sa javí ako nezávislý. Princíp difúzie, ktorý hovorí o tom, že kľúč a OT musia ovplyvniť čo najviac znakov v ZT (18).

7.3 Modul komponovania

Nasledujúce kapitoly budú tvoriť jadro riešenia našej problematiky. V nich vysvetlíme, aké prvky tvoria množinu **H** a aké procesy prebiehajú v moduloch **C_1** a **C_2**.

Vo všeobecnosti procesy transformujú vstup **I**, ktorý sa javí ako náhodný, tvorený prvkami množiny **T_2** na výstup **O** tvorený prvkami množiny **H** a naopak. Ak potrebujeme zo vstupu **I** odobrať **n** bitov, ako vstup pre konkrétnu operáciu, tento vstup zapíšeme ako **I(n)**.

V každej kapitole vysvetlíme kroky, ktoré daný proces vykonáva. Ku každému kroku definujeme inverzný krok. Taktiež spomenieme alternatívne riešenia a dôvody, prečo sme sa rozhodli ich nezvoliť. Následne formálne zobrazíme daný proces. Pri formálnom zobrazení sme sa inšpirovali Mealyho automatom (19) a uml diagramami. Zmeny, ktoré sme vo formálnom zobrazení potrebovali vykonať na dôkladný opis daných procesov sú nasledovné.

Obr. 9: Zmeny vo formálnom zobrazení

Hudobný rozsah, ktorý je modul **C₁** schopný vygenerovať ako koherentný celok je jedna **perióda**. Teda komponovanie je navrhnuté tak, aby sme do rozsahu jednej periódy vedeli vložiť určitý počet dát. Toto je veľmi dôležitý fakt, vďaka ktorému získame pri riešení problematiky určité pozitíva, ale aj obmedzenia. Komponovanie periódy sme hierarchicky rozdelili tak, že najprv určíme tóninu a akordy, potom rytmus pre melódiu a nakoniec vo vytvorenom rytme doplníme výšky tónov.

7.3.1 Generovanie tóniny

Prvým krokom v module **C₁** je generovanie tóniny, ktorej tóny budú použité v melódii periódy. Na základe vstupu vieme vygenerovať **8** rôznych tónin. Vstup **I(3)** prevedieme na decimálnu hodnotu a tá nám (automat) určí danú tóninu. Tónina je reprezentovaná pomocou číselnej hodnoty **t** z obrázka 6 v prvom riadku. Toto číslo je zároveň aj tonikou stupnice určujúcej danú tóninu. Stupnicu sme fixne zvolili pre celý systém ako **durovú**. Následne vypočítame súhrn všetkých nôt patriacich do tejto tóniny.

Nech množina **N** = {0, 1, … 127} je súhrn všetkých nôt v 12 notovom hudobnom systéme reprezentovaných číselne a **t** je tonika durovej stupnice. Potom množina **M ⊂ N** je tónina pre **t**, a jej prvky vieme vyjadriť ako:

```
m0=t
for i = 1 … 60
  if (i mod 7 == 2 && i mod 7 == 6)
        mi=mi-1+1
  else
        mi=mi-1+2
```

Funkcia vyjadrená pseudokódom je v diagrame označená **m(t)**. Funkcia **m(t)** je vytvorená na základe definície durových stupníc. Experimentálne sme určili, že 60 tónov je dostatočný počet na komponovanie hudby. Taktiež nám tento rozsah vytvoril hornú hranicu, nad ktorou sú už príliš vysoké tóny. Množina **M** je podmnožinou **H.**

7.3.2 Generovanie rámcových tónov

Aby skladba v poslucháčovi navodzovala estetický dojem, skladatelia používajú na rôznych miestach určité konkrétne tóny v závislosti od kadencie daného tónu. Výskumom sme prišli na to, že na konci periódy, pre zdôraznenie ukončenia skladby, sa často vyskytuje tonika. Uprostred periódy na konci prvej frázy to môže byť medianta, dominanta alebo subtonika danej stupnice. Pričom najmä subtonika vyjadruje pocit očakávania pokračovania. Určili sme, že dané dôležité tóny sa budú nachádzať v oktáve 5 alebo 6 a ich výšky sme vygenerovali pomocou výpočtu na množine **N.** Nazvime ich **rámcové tóny**.

```
subtonika1=t+59;
subtonika2=t+71;
medianta=t+64;
dominanta=t+67;
tonika1=t+60;
tonika2=t+72;
```

Tieto hodnoty patria do množiny **H**, pri komponovaní ich použijeme neskôr. Ak hodnoty pre určité konkrétne **t** presahovali oktávu 5 alebo 6, jednotlivo sme ich upravili. V diagrame tento krok nájdeme pod názvom *posledná + prostredná nota*.

Reverzný proces v module $\mathbf{C_2}$ bol nasledovný. Keďže tonika bola použitá ako posledná v perióde, z melódie sme ju extrahovali. Na základe porovnania s hodnotami tonika1 a tonika2, ktoré boli pre každú tóninu unikátne sme získali hodnotu **t**. Z hodnoty **t** následne získame množinu **M**, hodnoty ostatných rámcových tónov a vstup **I(3)**, na základe ktorého bola **t** vygenerovaná v module $\mathbf{C_1}$.

7.3.3 Generovanie akordických tónov a progresie akordov

Množstvo moderných piesní používa v melódii z pravidla 4 rôzne akordy. Ich poradie v piesni tiež nie je jednotné. Komponovanie využívajúce tento model sme navrhli práve v tejto sekcii. Najprv sme zvolili 4 akordy pre celú periódu. Potom sme vygenerovali jednotlivé tóny patriace akordom a progresiu, v ktorej budú v perióde zahrané. Dané akordy, ktoré sme zvolili boli kvintakordy **I V VI IV** pre durovú stupnicu **t**. Pre jednoduchosť práce ich môžeme reprezentovať množinou **{0,1,2,3}**.

Počet akordických tónov pre každý akord sme zvolili 4, je to znova kvôli tomu, že v neskoršom rozhodnutí, ktorý z nich použijeme vieme uložiť 2 bity informácie. Označíme ich {a,b,c,d}. Výpočet ich výšok je podobný ako v predošlej sekcii pri rámcových tónoch.

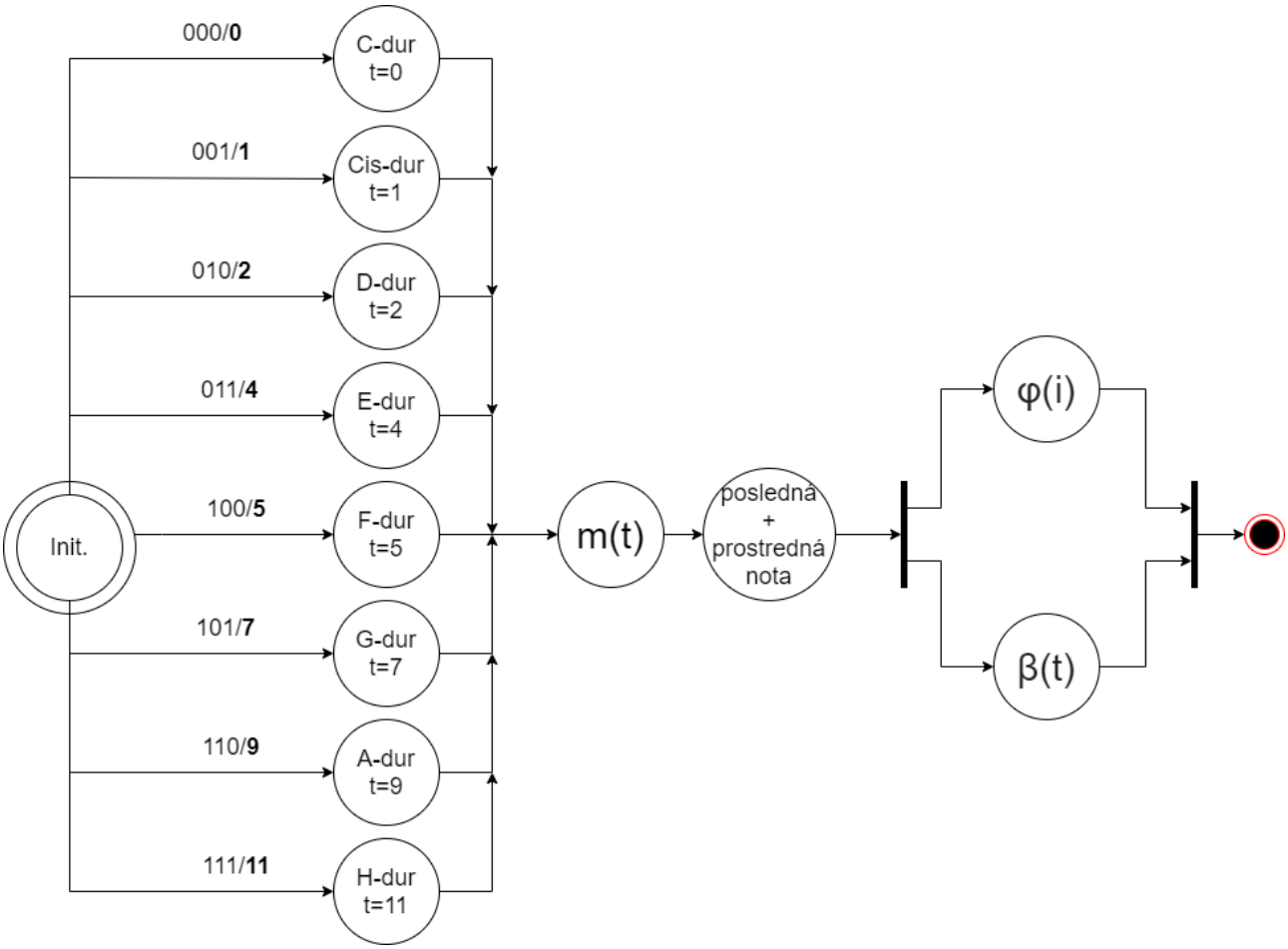
```
Akord 0 = {t + 55, t + 60, t + 64, t + 67}
Akord 1 = {t + 62, t + 67, t + 71, t + 74}
Akord 2 = {t + 52, t + 57, t + 60, t + 64}
Akord 3 = {t + 60, t + 65, t + 69, t + 72}
```

Pozorný čitateľ si všimne, že kvintakord v hudbe je tvorený 3 rôznymi tónmi. Teda čo tvorí zvyšný tón? Štvrtý tón v týchto množinách je ten istý, ako prvý tón, len o oktávu vyššie. Môžeme si všimnúť, že relatívna vzdialenosť medzi nimi je 12 čo je interval jednej oktávy. Tento proces môžeme v diagrame vidieť pod označením **β(t)**.

Progresiu akordov určíme tak, že konkrétny akord môže byť v perióde použitý iba raz a použité sú všetky. Potom progresia akordov môže byť vyjadrená ako určitá **permutácia** na množine **{0,1,2,3}**. Počet rôznych permutácií na 4 prvkovej množine je 4! = 24. Teda máme 24 možností pre rôzne usporiadanie akordov. Kvôli práci s binárnym reťazcom si znovu zvolíme 16 možností. Rozhodnutie, ktorú permutáciu použijeme prebieha nasledovne. Vstup **I(4)** prevedieme na decimálny tvar. To číslo potom vyjadruje poradové číslo v lexikograficky usporiadanom zozname permutácií (1). Teda pre progresiu akordov použijeme permutáciu na tom mieste v usporiadanom zozname. Napríklad ak vstup **I(4)** je **0000**, tak permutácia je {0,1,2,3}, alebo ak je vstup **1111** tak permutácia je {2,1,3,0}. Tento proces je v diagrame označený ako **φ(i)**.

Akordické tóny aj progresia akordov patria do množiny **H**. Sú dva spôsoby, ako ich pri komponovaní využijeme. Jeden z nich opíšeme v sekcii o melódii. Druhým je použitie vygenerovaných kvintakordov v basovej linke. Periódu rozdelíme na 4 časti. Každá časť má rozmedzie 2 taktov a je v nej hraný príslušný akord na základe permutácie. Tóny hraného kvintakordu sa budú nachádzať v oktáve 3 alebo 4. Tento krok je potrebný na spätné získanie progresie akordov v module $\mathbf{C_2}$. V reverznom kroku extrahujeme dané akordy z basovej linky. Keďže poznáme **t**, porovnávaním akordických tónov získame potrebnú permuáciu. Potom ju porovnáme s usporiadaným zoznamom a poradové číslo prevedené na binárne bude tvoriť vstup **I(4)**.

Alternatívne kroky, ktoré sme zvažovali boli nasledovné. Využitie molových tónin, nie len durových. Vygenerovanie väčšieho množstva akordických tónov alebo rámcových tónov. Zvolenie väčšieho množstva akordov a tým pádom aj komplexnejšie progresie. Tieto kroky sme nezvolili preto, lebo nám neponúkajú nový inovatívny prístup na riešenie tejto problematiky. Iba rozširujú to čo máme otestované a vieme, že funguje. Samozrejme, nebol by problém podobné výpočty aplikovať aj na tieto alternatívy. Ostáva to ako otvorená téma do budúcna.



Obr. 10: Schéma generovania harmónie

7.3.4 Generovanie rytmu

Čo sa rytmického rozloženia periódy týka, rozhodli sme sa, že použijeme 32 dobovú formu. Teda perióda je rozdelená na 8 taktov a každý takt na 4 doby. Dĺžku jednej doby sme určili ako štvrťovú, teda takty v perióde sú štvor-štvrťové. Samotná doba sa dá rozložiť ešte na rôzne **rytmické skupiny**, podľa toho aké dĺžky jednotlivých nôt sa v dobe nachádzajú. Práve rozhodnutie, ktorú konkrétnu rytmickú skupinu pre danú dobu použijeme, určujeme na základe vstupu **I**. Na základe experimentov sme sa rozhodli, že počet použiteľných rytmických skupín bude **8** a vstup do operácie rozhodovania je **I(3)**. V tomto kroku sme taktiež zohľadnili, na ktorých miestach v dobe bude **pomlčka** teda melódia nebude hrať. Tento proces môžeme vidieť v diagrame 12 nasledovne. Výstupy v prechodoch sú rytmické skupiny v názvosloví knižnice JFugue. Názvy stavov určujú pomer, ktorým dĺžky jednotlivých nôt dobu rozdeľujú.

7.3.4.1 Problém s rytmickou pulzáciou

Pravdepodobnosť, že konkrétna rytmická skupina bude použitá v danej dobe je **1/8**. Pri 32 dobách nám táto pravdepodobnosť hovorí o tom, že každú skupinu bude možné počuť približne 4 krát v celej perióde. Týmto do istej miery zabezpečíme určitú formu rytmickej pulzácie. Experimentami sme ale zistili, že estetická stránka rytmickej pulzácie nie je dosiahnuteľná týmto náhodným rozložením daných rytmických skupín. Taktiež na miestach v skladbe, kde tóny majú signifikantnú kadenciu je potrebné, aby dĺžka tónu bola rovná celej dobe. Opakovanie rytmických skupín dôb vo vnútri taktu sme nazvali **interné** opakovanie. Opakovanie rytmu v dvoch rôznych taktoch sme nazvali **externé** opakovanie. V rytmickej pulzácii by sa mali vyskytovať obe uvedené formy opakovania rytmu.

Tento problém sme vyriešili tak, že namiesto 32 rytmických skupín pre periódu, sa vygeneroval len určitý **c** počet. Hodnotu **c** sme na základe pokusov nastavili na 12. Vygenerovaným rytmickým skupinám sme pridelili index. Potom sme ich deterministicky rozložili na konkrétne miesta v celej perióde, tak aby bolo dodržané interné a externé opakovanie rytmu. Týmto spôsobom sa znížil počet bitov, ktorý je možné uložiť v rytme hudby, ale zvýšila sa estetická kvalita skladby. Funkcia zodpovedná za tieto operácie je v diagrame označená **ρ(c).** Funkcia **ρ(c)** vygeneruje určité jednorozmerné pole **R**, ktoré hovorí o tom, aká rytmická skupina bude použitá v konkrétnych dobách a aké majú jednotlivé tóny dĺžky. Pole **R** patrí do množiny **H**.

Perióda																
Fráza 1																
Doba	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
c	"q"	0	1	0	2	3	4	5	"q"	0	1	0	6	7	8	"q"
Fráza 2																
Doba	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
c	9	10	9	11	2	3	4	5	9	10	9	11	6	7	8	"q"

Obr. 11: Rozloženie jednotlivých rytmických skupín v perióde

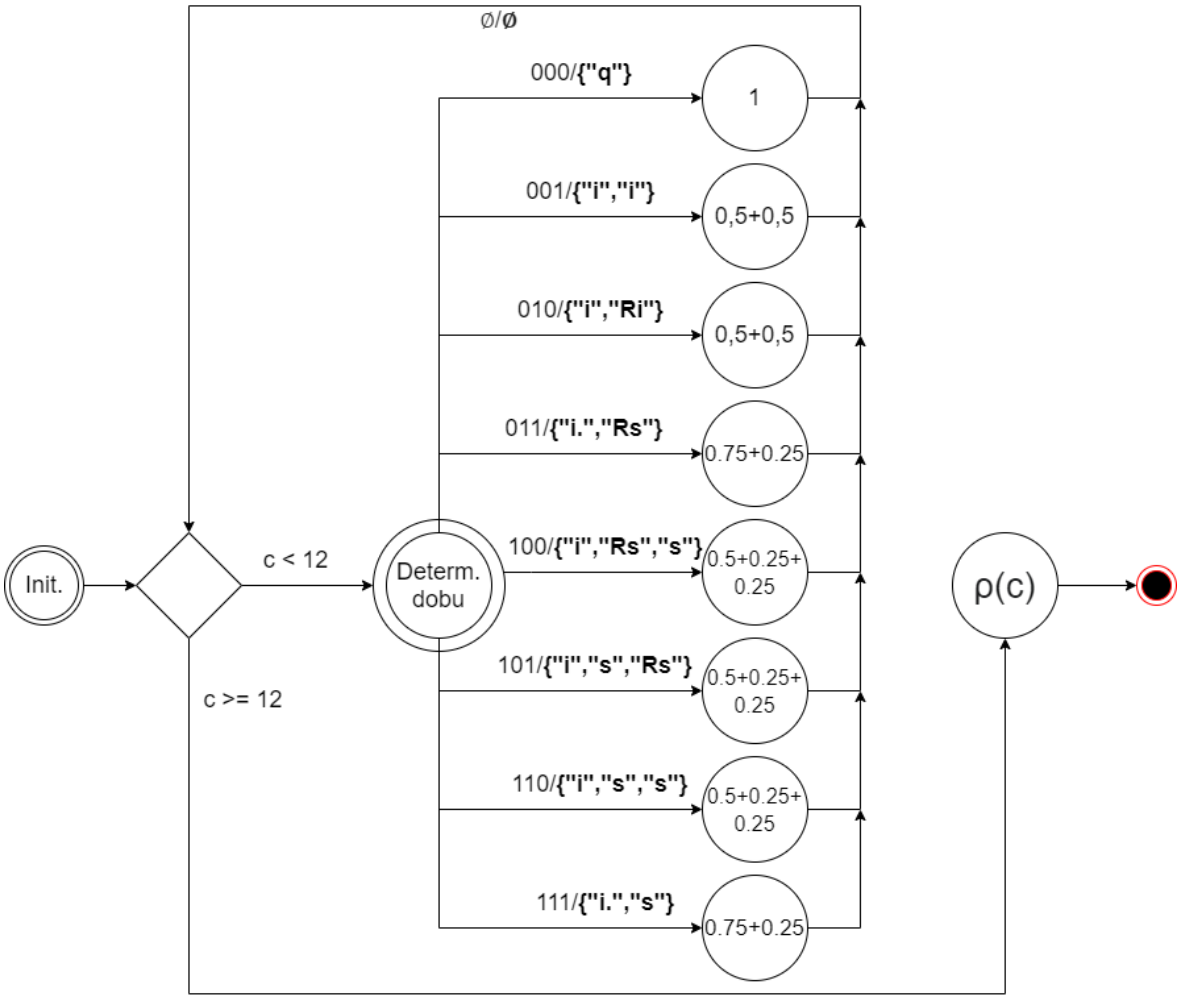
Reverzný postup, ako sme z hudby dostali rytmus **R** a vstup **I** je nasledovný. Hodnotu dĺžky tónu vieme získať vďaka tomu, že štandard Jfugue ho zapisuje ako suffix za hodnotu výšky tónu. Prechodom a ukladaním suffixov získame rytmické označenie všetkých tónov periódy. Získanie konkrétnych rytmických skupín v dobách je vyjadrené týmto pseudokódom.

Nech **x** je numerická hodnota a **n** názov označenia dĺžky tónu. **s** je súčet dĺžok tónov v dobe. **r** je rytmická skupina tvorená názvami označení dĺžok. Potom:

```
s=0
while not end
  if(s == 1)
       return r
  else
       ADD n to r
       s = s + x_n
```

Z takto získaného poľa vytvoríme pole **R** a na základe tabuľky 6 získať podľa indexu, ktoré konkrétne rytmické skupiny boli generované na základe vstupu. Reverzným procesom konečného automatu získame decimálne hodnoty pre vstupy. Tie prevedieme na binárne hodnoty, zoradíme a dostaneme prvotný vstup **I(3)** * **c**.

Alternatívne riešenie je stále otvorené pri návrhu funkcie **ρ(c)**, kde je pravdepodobne možné získať lepšie znejúcu rytmickú pulzáciu. To, ktoré v práci uvádzame je výsledkom časovo obmedzeného počtu experimentov. Rovnako sme experimentovali aj s počtom možných rytmických skupín pre 1 dobu. **16** sme zamietli kvôli tomu, že pre vyplnenie všetkých možností sme museli použiť nepravidelné rytmické skupiny, ktorých názvy v práci neuvádzame. Ak ich znenie bolo v perióde použité, bolo potrebné pre zachovanie estetického dojmu dané skupiny pridať deterministicky na iných miestach, čo radikálne znížilo hodnotu **c** a tým pádom aj počet bitov na vstupe **I**. Rytmus tvorený **4** rôznymi rytmickými skupinami znel príliš jednoducho a taktiež znižoval **I** z iných dôvodov.

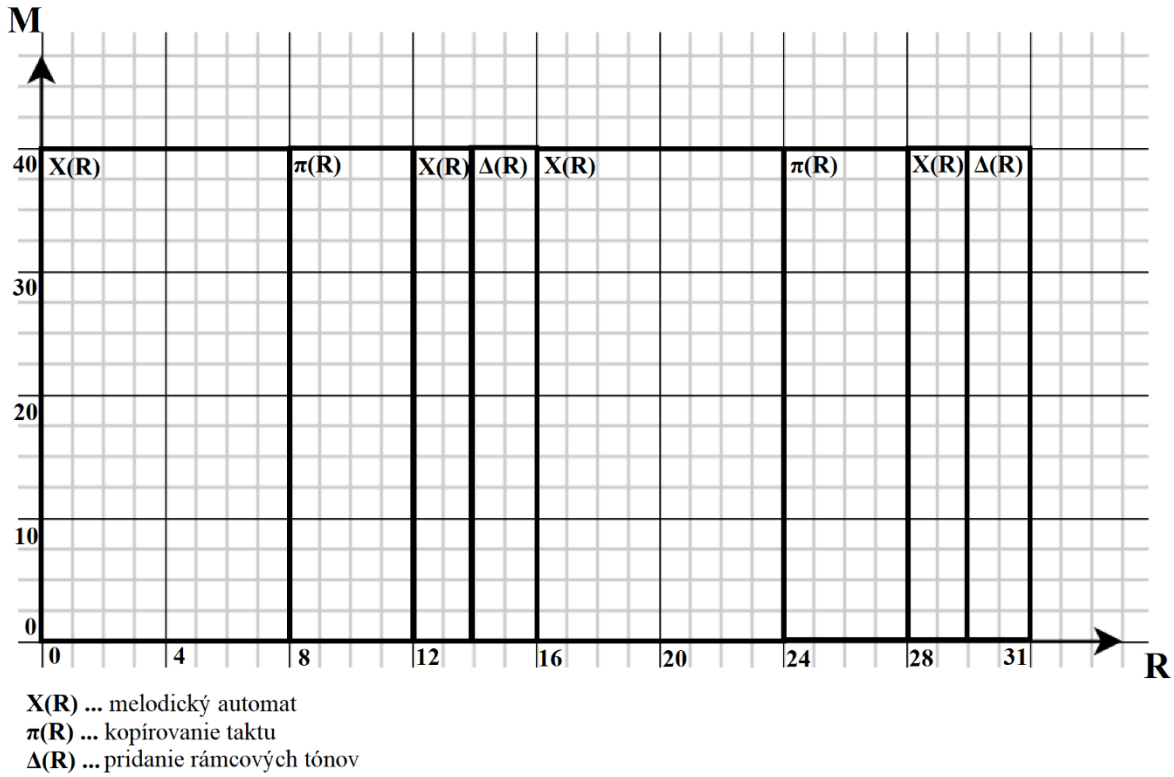


Obr. 12: Schéma generovania rytmu

7.3.5 Generovanie melódie

Teraz prichádzame k sekcii, ktorej vysvetlime ako sme schopní uložiť najväčší počet dát a zároveň vytvoriť v perióde melódiu s estetickým dojmom. Zosumarizujme si aké elementy máme vygenerované a môžeme s nimi pracovať. Všetky výšky tónov tóniny **M**, dĺžky tónov obsiahnuté v **R**, progresiu akordov, akordické tóny daných akordov a rámcové tóny. Pre zjednodušenie predstavy si môžeme melódiu zobraziť ako diagram závislosti výšky tónu z množiny **M** od časového priebehu periódy vyjadrenej poľom **R**. Ďalej si definujeme premenné, ktoré nám budú hovoriť o tom s akou hodnotou z týchto množín pracujeme. Premenná **x** vyjadruje výšku tónu na množine **M**. Premenná **beatCounter** hovorí o tom, v ktorej dobe sa momentálne nachádzame v poli **R** a premenná **noteCounter**, pre konkrétny tón v dobe.

Komponovanie melódie sme rozdelili na tri rozdielne algoritmy, ktoré sú volané na rôznych miestach v perióde. Ich označenia a intervaly v poli **R** kedy sú volané je možné vidieť v diagrame 13. Jednotlivé algoritmy si teraz priblížime.



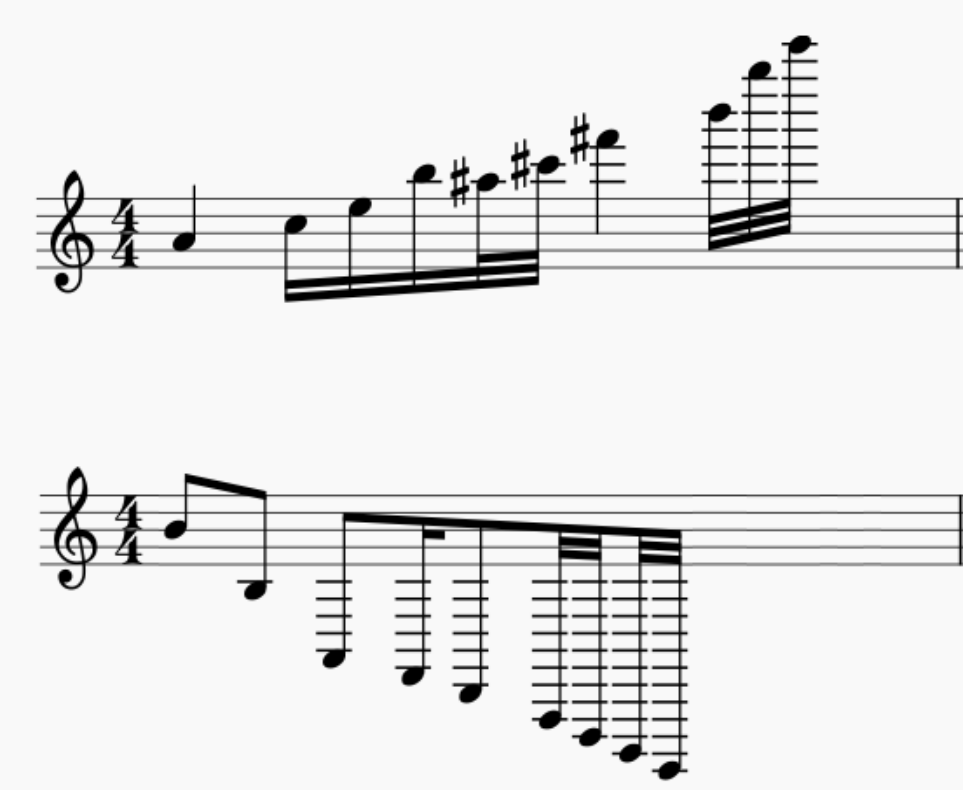
Obr. 13: Diagram závislosti výšky tónu od rytmu v perióde. Rozdelenie na sekcie.

7.3.5.1 Melodický automat X(R)

Tento algoritmus vie reprezentovať najväčší počet bitov zo vstupu **I.** Jeho formálna reprezentácia je vidieť v diagrame číslo 15. Algoritmus začína v inicializačnom stave Init, keď beatCounter dosiahne hodnotu **z**, algoritmus sa ukončí.

Prvé rozhodnutie hovorí o tom, či sa nachádzame na mieste v perióde, kde pre zachovanie harmónie musí byť vygenerovaný **akordický tón**. Z hudobnej praxe vieme, že by to mal byť prvý tón v prvej a tretej dobe taktu. Túto pozíciu zistíme pomocou premenných beatCounter a noteCounter. Následne transformáciou vstupu **I(2)** vygenerujeme konkrétny akordický tón prislúchajúci akordu, ktorý je hraný v takte, v ktorom sa nachádzame. Výška tónu sa vloží do **x** a vypíše do výstupu **O** s dĺžkou získanou z poľa **R**.

Ak nie sme v pozícii, kedy generujeme akordický tón, pristupujeme ku komponovaniu melódie na základe zmien v melodickom pohybe. Tieto zmeny vytvárame transformáciou vstupu **I**. Algoritmus najprv skontroluje, či sa hodnota **x** nachádza v intervale **J.** Ktorý nám určuje rozmedzie tónov, ktorých výška nie je príliš vysoká alebo nízka. Hraničné hodnoty tohoto intervalu určujú premenné $\mathbf{j_1}$ a $\mathbf{j_2}$. Ak by tento interval v algoritme nebol, mohli by v melodickom pohybe nastať 2 nežiadúce situácie zobrazené na obrázku 14.



Obr. 14: Nežiadúci melodický pohyb bez ohraničujúceho intervalu

Ak je hodnota **x** menšia ako $\mathbf{j_1}$ nasledujúci tón bude vyšší, ak bude väčšia ako $\mathbf{j_2}$ tón bude nižší. Ak bude v intervale **J** o stúpaní melodického pohybu sa rozhodne na základe vstupu **I(1)** tak ako je zobrazené v automate. Ďalším krokom je rozhodnutie na základe **I(1)** či v melodickom pohybe nasledujúca nota bude **step** alebo **leap.**

1. Ak sa rozhodne **step** a predošlým stavom bolo Stúpanie do výstupu **O** sa vloží hodnota **x+1.** Ak ním bolo klesanie, tak **x-1**. Ako suffix za túto hodnotu vložíme označenie dĺžky tónu z **R**. Algoritmus sa vráti do Init.
2. Ak sa rozhodne **leap**, pomocná premenná **k** sa nastaví na hodnotu 1 ak sme v stúpaní, ak v klesaní tak na -1. Po tomto rozhodnutí nasleduje ďalšie so vstupom **I(2)**, ktoré určí s akým intervalom nastane daný leap. Konkrétne hodnoty, ktoré sa vypíšu do výstupu **O** je možné vidieť v automate. Vo výpise taktiež nastavíme dĺžku tónu z **R**. Po tomto kroku sa algoritmus vráti do Init.

Zmeny v melodickom pohybe patria rovnako do množiny **H.** Keďže algoritmus je možné reprezentovať pomocou Mealyho automatu, hodnoty vstupu **I** v module $\mathbf{C_2}$ získame pomocou reverzného procesu tohoto automatu.

7.3.5.2 Kopírovanie taktu π(R)

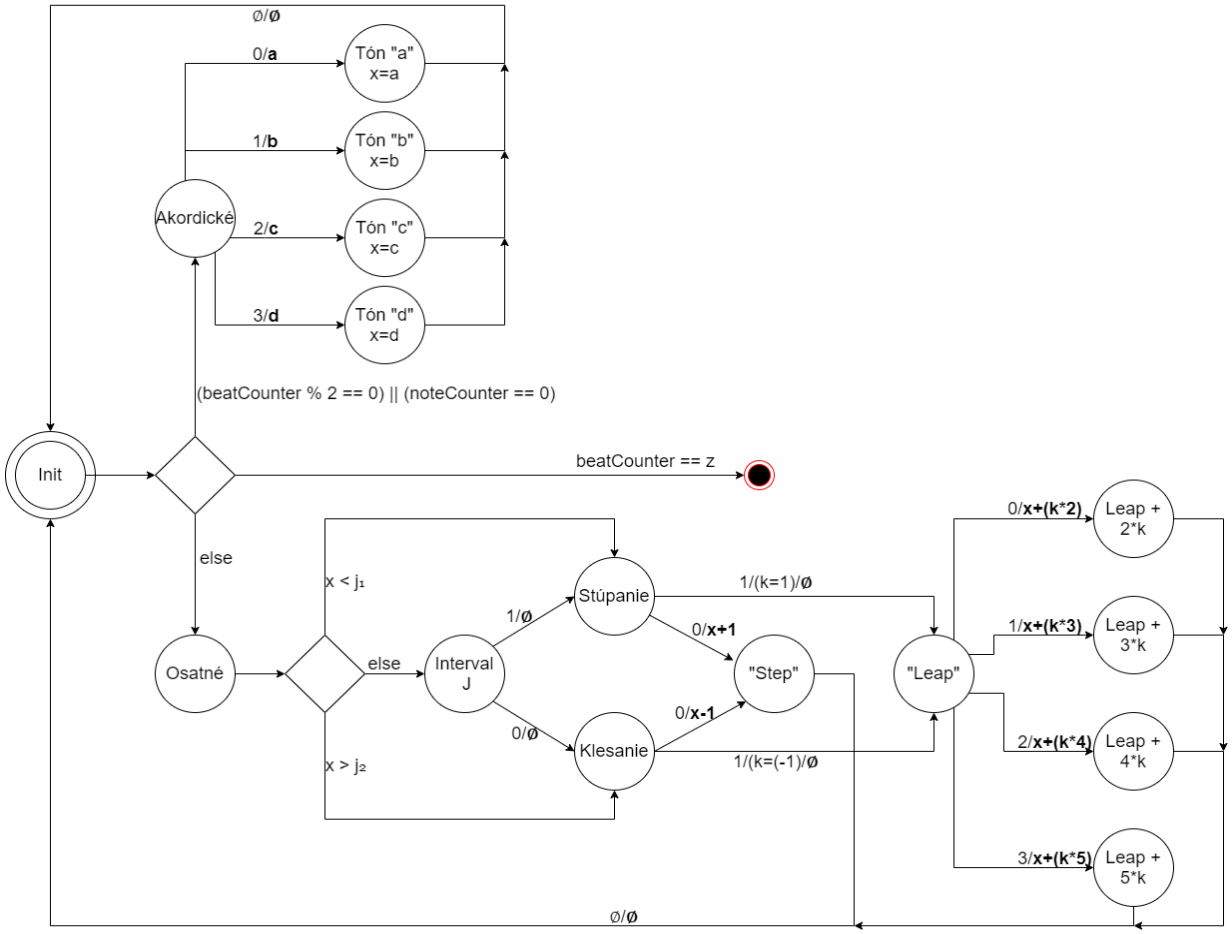
Z výskumu (20) a na základe experimentov sme prišli k záveru, že v melódii kvôli zachovaniu estetického dojmu je dôležité, aby sa niektoré sekcie opakovali. Exaktné opakovanie tónov nezabezpečuje tento dojem. Pri kopírovaní sekcie tónov musíme zohľadniť fakt, že takt, do ktorého danú sekciu kopírujeme môže byť v inom akorde. Kvôli jednoduchosti práce sme sa rozhodli že celky, ktoré sa budú v perióde opakovať budú mať rozsah jedného taktu. V periódach opakujeme obsah 1. taktu v 3. takte a 5. taktu v 7. takte. Zmenu, ktorú sme museli vykonať pri kopírovaní týchto taktov vieme vyjadriť nasledovne. Ak je tón v takte **A** basovým tónom akordu, ktorý v **A** hráme, tak v takte **B** pri kopírovaní taktov musí tento tón zmeniť svoju hodnotu na basový tón v akorde, v ktorom je hraný takt **B**. Tento vzťah vieme numericky vypočítať a dokonca určiť na základe vstupu **I(1)** či pôjde o basový tón v danej oktáve, alebo o oktávu nižšie. Pomocou tohto výpočtu získame určitý **offset**, podľa ktorého zmeníme výšky všetkých tónov kopírovaného taktu.

Reverzný krok je pomerne jednoduché spraviť. Na základe rozdielu medzi prvým tónom originálneho taktu a prvým tónom skopírovaného taktu vieme určiť offset. Na základe ktorého získame hodnotu **I(1)**. Hodnoty zvyšných tónov môžeme preskočiť.

7.3.5.3 Pridanie rámcových tónov Δ(R)

Konkrétne miesta, kde sa v perióde vyskytnú rámcové tóny máme zadefinované. Keď sa beatCounter dostane na pozíciu na konci prvej frázy, zo vstupu **I(2)** sa rozhodne, ktorý z tónov {subtonika1, subtonika2, medianta, dominanta} použijeme. Ak je algoritmus na konci periódy, na základe **I(1)** rozhodne, ktorý tón použije z {tonika1, tonika2}. Môže nastať situácia, kedy hodnota **x** je príliš nízka alebo vysoká a interval medzi **x** a pridaným rámcovým tónom je natoľko veľký, že by leap v melodickom pohybe nebol estetický. Tento problém sme vyriešili tak, že doba predchádzajúca tej, v ktorej je rámcový tón má za úlohu priblížiť melodický pohyb k danému tónu. Vypočíta sa rozdiel medzi **x** a rámcovým tónom. Na základe rozdielu sa potom predchádzajúca doba vyplní stúpajúcimi alebo klesajúcimi steps.

Reverzný proces najprv ignoruje predchádzajúcu dobu, keďže tóny v nej nezískavame na základe vstupu. Potom pomocou porovnávania získaného rámcového tónu vieme určiť, ktorý vstup **I(2)** sme použili na konci prvej frázy, a ktorý vstup **I(1)** na konci periódy.



Obr. 15: Schéma generovania melódie

7.4 Zhrnutie

Vstup **I** javí prvky náhodnosti, nakoľko je výsledkom operácie AES. Hudba výlučne generovaná z tohto vstupu pomocou melodického automatu je nepredvídateľná a často neestetická. Tento nedostatok riešime pomocou opakovania sekcií, pridávania rámcových tónov alebo funkcie **ρ(c)** v rytme. Tieto riešenia majú za následok zníženie možného rozsahu dát, ktoré môžeme pomocou hudby reprezentovať. Na základe tejto problematiky nám vznikajú dva súperiace faktory, ktoré navzájom ovplyvňujú naše riešenie. Aspekt rozsahu dát a aspekt estetiky hudby. Ak si čitateľ spomenie, týmto vzťahom sa zaoberal už J. C. Ortiz pri svojom hudobnom kryptograme.

Alternatívne riešenie spočíva v tom, že medzi týmito aspektmi nájdeme určité equilibrium, pri ktorom zabezpečíme estetickú kvalitu a zároveň užitočný rozsah bitov pre prenos informácie. Riešenie, ktoré máme k dispozícii je výsledkom časovo obmedzených experimentov.

Aký minimálny a maximálny rozsah bitov pre vstup **I** má naše riešenie? Takmer všetky algoritmy komponovania pracujú so stabilným rozsahom bitov pre vstup. Výnimkou je melodický automat, ktorého veľkosť vstupu závisí od toho koľko tónov je v perióde a či sa vygeneruje step alebo leap. Veľkosť vstupu **I** je:

1. Minimálna:
    a. Stabilná časť: **66 bitov**.
    b. **X(R)** s 1 tónom v dobe a všetky tóny steps: **10 bitov**
    c. Rozsah spolu pre periódu: **76 bitov**
2. Maximálna:
    a. Stabilná časť: **66 bitov**.
    b. **X(R)** s 3 tónmi v dobe a všetky tóny leaps: **150 bitov**
    c. Rozsah spolu pre periódu: **216 bitov**
3. Priemerná:
    a. Počas mesiacov experimentov sa priemerná hodnota pohybovala v rozmedzí **90 - 130 bitov** pre periódu.

V rozmedzí celej piesne sme schopní vygenerovať 5 unikátnych periód. Jednu pre intro, outro, verš, refrén a bridge. Z estetického hľadiska je perióda, ktorú vytvárame uvedeným spôsobom najviac podobná **veršu**. Pri tvorení periód pre jednotlivé sekcie je potrebné pozmeniť niektoré faktory komponovania. To nám môže taktiež negatívne ovplyvniť rozsah vstupu. Konkrétne zmeny potrebné pri komponovaní jednotlivých sekcií v práci neuvádzame.

8 Prieskumná časť

V kapitole X sme definovali, aké hudobné pravidlá musíme dodržať, aby bola hudba komponovaná našim systémom čo najviac príjemná na počúvanie, poprípade nerozoznateľná od bežnej, širokou verejnosťou počúvanej hudby. Na to, aby sme vedeli potvrdiť alebo zamietnuť, či bola táto podmienka splnená, sme vykonali prieskum názoru širokej verejnosti.

Hlavným cieľom prieskumu bolo určiť, či je respondent schopný rozoznať skladby vytvorené našim systémom od skladieb vytvorených človekom. Vedľajším cieľom bolo zistiť v akom tempe a pri použití akého hudobného nástroja je toto rozhodnutie pre respondenta náročnejšie.

Formou videa sme respondentovi sme predložili 8 nahrávok z rôznych skladieb. Ich MIDI reprezentácia bola prevedená do zvukových nahrávok pomocou programu MuseScore 3 (21). Tri skladby boli vytvorené našim systémom. Toto sú ich špecifikácie:

1. Skladba bola zahraná na organe, obsahovala basovú linku a tempo bolo 90 bpm.
2. Skladba bola zahraná na harfe, neobsahovala basovú linku a tempo bolo 120 bmp.
3. Skladba bola zahraná na klavíri, neobsahovala basovú linku a tempo bolo 90 bpm.

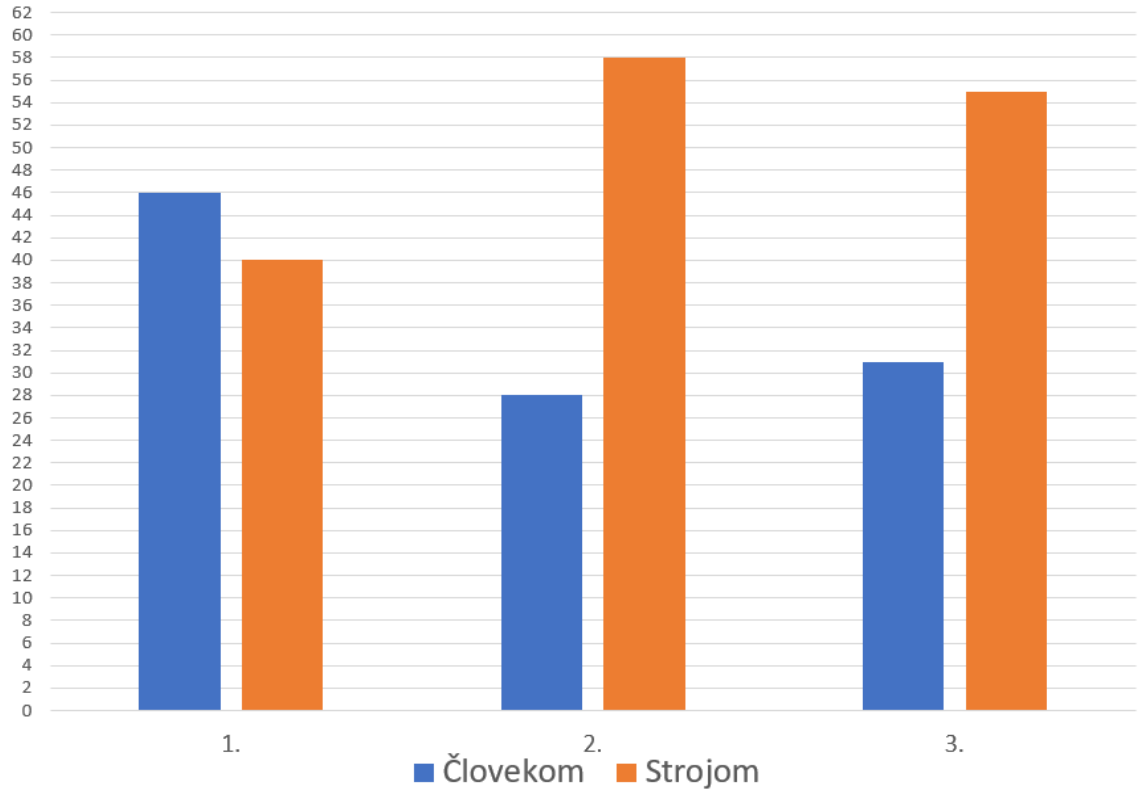
Ďalších 5 nahrávok tvorili výňatky z rôznych skladieb vytvorených skladateľmi, alebo autormi hudobných doprovodov. Dĺžka výňatkov sa zhodovala s dĺžkou skladieb vytvorených našim systémom. Názvy skladieb sú:

4. Jamie Henke - How to Write a Melody? (20)
5. Grant Kirkhope, Eveline Novakovic - Fungi forest daytime
6. Atsushi Chikuma, Tomoyuki Hamada - Level ok
7. Dmitrij D. Šostakovič - Fúga v A dure
8. Igor F. Stravinskij - Svätenie jari

Respondentov sme sa na konci prieskumu spýtali dodatočnú otázku, či si rozhodnutím boli istí, alebo pri niektorých skladbách váhali. Prieskum prebiehal v intervale jedného týždňa a počet respondentov bol 86. Respondenti patrili medzi širokú nešpecifikovanú verejnosť.

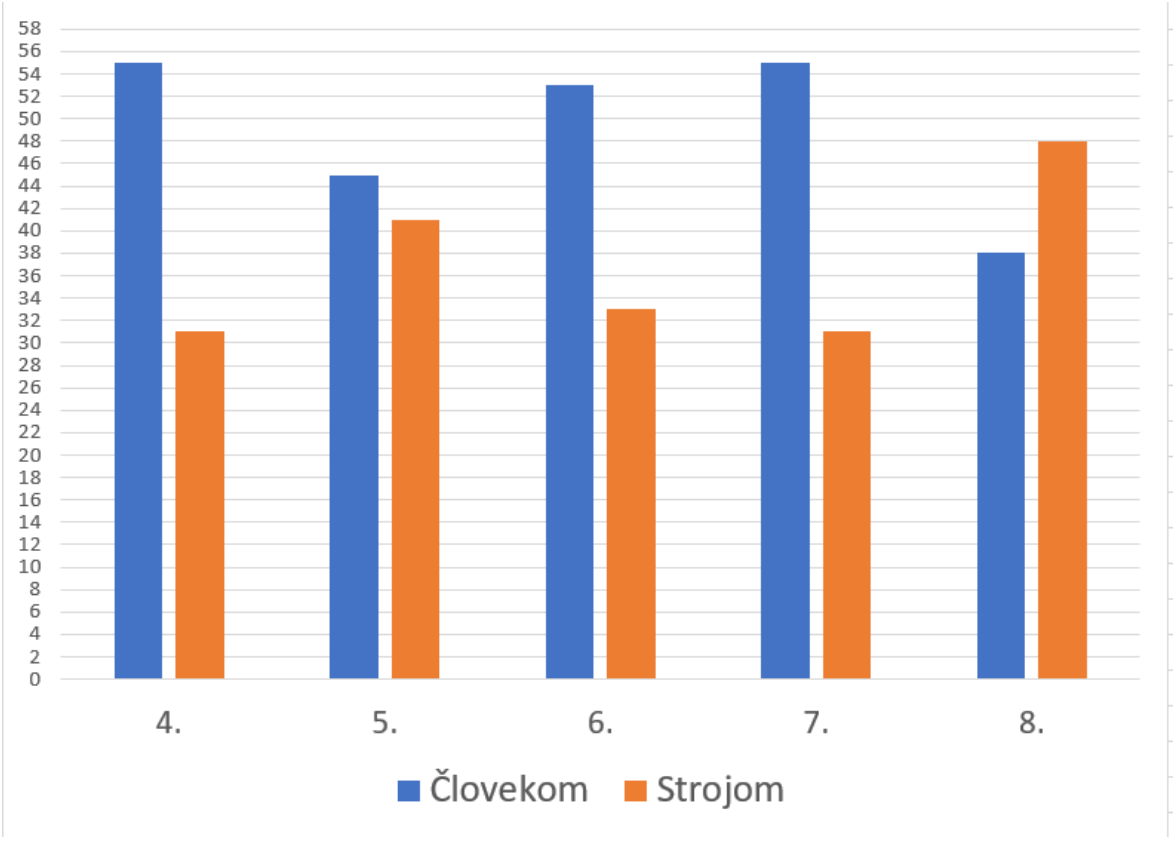
8.2 Spracovanie prieskumu

Výsledky prieskumu sme spracovali graficky pomocou diagramov. Najdôležitejším skúmaným údajom je pre nás pomer, ktorý vyjadruje koľko respondentov určilo skladby vygenerované našim systémom ako vytvorené človekom a koľko strojom. V diagrame č. 16 je možné vidieť tento pomer pre jednotlivé skladby vytvorené kryptosystémom.



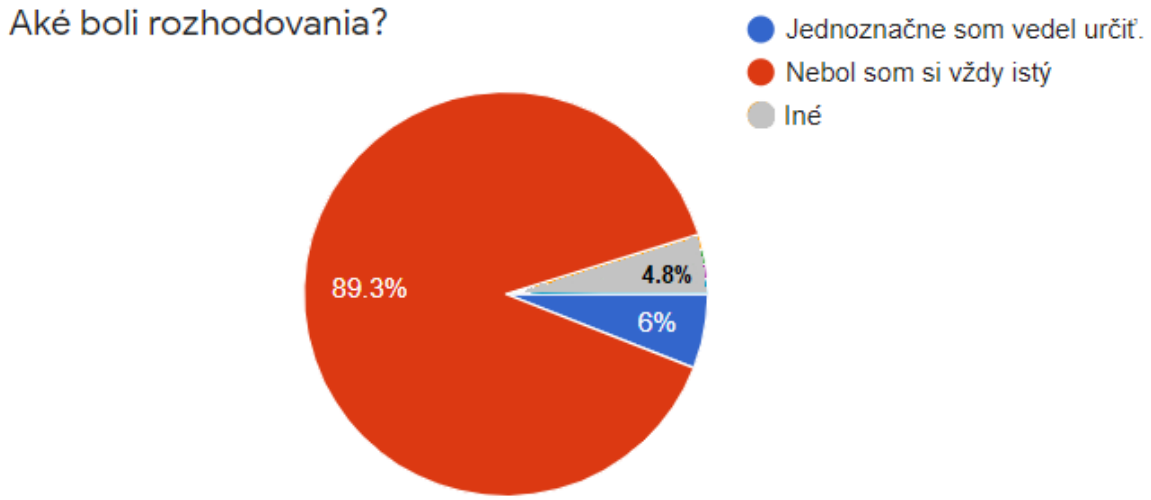
Obr. 16: Počet správnych a nesprávnych odpovedí pre konkrétne piesne

V diagrame 17 je možné vidieť odpovede pre ostatné skladby vytvorené človekom.



Obr. 17: Počet správnych a nesprávnych odpovedí pre priesne vytvorené človekom

Ďalším údajom v diagrame 18 je odpoveď na otázku, či si respondenti boli istí svojimi odpoveďami, alebo pri rozhodovaní neboli istí a tipovali.



Obr. 18: Subjektívny názor respondentov

8.3 Vyhodnotenie prieskumu

Pri tomto prieskume vyšlo najavo, že respondent nie je vždy schopný vydedukovať, či je skladba vytvorená človekom alebo automaticky našim kryptosystémom. Dôležitým zistením bolo, že úspešné oklamanie poslucháča skladby závisí tak od hudobných pravidiel, ako aj od nástroja, ktorý tieto skladby hrá a od tempa, v ktorom je skladba. Pravdepodobným dôvodom tejto závislosti je skutočnosť, že poslucháč nie je naučený prijímať komplexnejšie, rýchlejšie melódie od všetkých hudobných nástrojov. Preto je pre pokračovanie na tejto práci dobré preskúmať, ktoré hudobné nástroje sú vhodné pre konkrétny typ melódie a tempa.

9 Odporúčania pre prácu v budúcnosti

Uvedomujeme si, že napriek tomu, že náš kryptosystém do istej miery spĺňa ciele, ktoré sme preň určili, existuje ešte mnoho spôsobov, ktorými by sa dal vylepšiť. V tejto kapitole spíšeme myšlienky, ktoré by v budúcnosti mohli prispieť k jeho rozvoju.

Predpokladáme, že po kryptografickej stránke je práca dostatočne rozvinutá a tým pádom je možné venovať sa iným oblastiam. V prvom rade by bolo vhodné zvýšiť estetickú stránku generovanej hudby. Hudba by mohla mať formu motivickej práce (22), kde by bol vytvorený výrazný, pomerne ucelený hudobný úryvok, ktorý by slúžil ako motív. Jeho opakovanie a modifikácia by mohla v skladbe vytvoriť sekcie, v ktorých vzniká napätie a sekcie, v ktorých sa napätie uvoľňuje. Tento postup by mohol viesť k tvorbe rôznych hudobných viet, líšiacich sa od striktnej frázy a periódy, ktoré používame. Pre celkové zvýšenie hudobnej kvality odporúčame preskúmať všetky inštrukcie, ktoré poskytuje MIDI štandard. Ich použitie na základe vstupu by mohlo taktiež zvýšiť celkový bitový rozsah. Každý MIDI kanál môže byť použitý na zakódovanie iného hudobného nástroja, ktorý môže slúžiť ako melodická basa, rytmický doprovod atď. Skladby by mohli byť vytvorené v iných hudobných žánroch, vďaka čomu by sme získali väčšiu rôznorodosť generovanej hudby. Odporúčame preskúmať podžánre popovej hudby ako jazz, rock, blues. Alebo využiť iný hudobný systém ako európsky napríklad čínsky, arabský alebo mikrotonálny. Keďže manuálny postup vytvárania procesov, ktoré z hudby získavajú bitový vstup je časovo náročný, odporúčame modelovať veľkú časť procesov pomocou konečných automatov, pre ktoré je jednoduchšie vytvoriť reverzné procesy.

Záver

Na začiatku sme si ukázali, akým spôsobom sa hudba využívala pre kryptografické účely v minulosti. Tento výskum nám slúžil ako odrazový bod pri návrhu nášho kryptosystému. Ďalej sme čitateľa voviedli tak do názvoslovia a vzťahov využívaných v hudobnej teórii, ako aj do ich reprezentácie v knižnici JFugue. Pre usmernenie tvorenia kryptosystému sme definovali hudobné a kryptografické požiadavky, ktoré by mal spĺňať. Podrobnejšie sme spomenuli inkrementálny postup návrhu systému, problémy, s ktorými sme sa stretli a k nim príslušné riešenia. Jadro práce tvoril návrh algoritmov zodpovedných za komponovanie hudby na základe binárneho reťazca. Spolu s návrhmi sme predložili aj postupy, ako binárny vstup z hudby získať späť. Taktiež sme spomenuli alternatívy, ktoré sme zvažovali počas navrhovania. Súčasťou práce bola aj prieskumná časť, na základe ktorej sme zhodnotili, že kryptosystém je do určitej miery možné použiť na prenos utajenej informácie. Zároveň sme prišli k záveru, že v práci je priestor pre zlepšenie, ktorému sa venuje kapitola č. 9.

Najdôležitejším prínosom tejto práce je dôkaz, že idea vytvorenia kryptosystému schopného vytvárať hudbu na základe utajenej správy je realizovateľná.

Súčasnú implementáciu s názvom Musipher, naprogramovanom v jazyku Java, je možné vidieť v tomto repozitári (23).

Zoznam použitej literatúry

1. **GROŠEK, O. -- VOJVODA, M. -- ZAJAC, P.** *Klasické šifry.* Bratislava : Vydavateľsvo STU, 2007. ISBN 978-80-227-2653-5.
2. **IAIN, Fenlon.** *The Renaissance: From the 1470s to the end of the 16th century.* s.l. : Springer, 1990. ISBN 978-13-492-0536-3.
3. **MARY, Simoni.** Algorithmic Composition: A Gentle Introduction to Music Composition Using Common LISP and Common Music. [Online] https://quod.lib.umich.edu/s/spobooks/bbv9810.0001.001/1:5/--algorithmic-composition-a-gentle-introduction-to-music?rgn=div1;view=fulltext&fbclid=IwAR29eQUyatiOPhteUD74E49OLD4lZgQfkK_-ZNA-LfDQwHWkf-2lfxOZjWg.
4. **PAUL, Marks.** Hiding Data in Music. [Online] 26. 3 2019. https://cacm.acm.org/news/235722-hiding-data-in-music/fulltext.
5. **Musical Cryptogram - Wikipedia.** [Online] 2020. https://en.wikipedia.org/wiki/Musical_cryptogram.
6. **GASPAR, Schott.** *Schola Steganographica.* s.l. : Jobus Hertz, 1680.
7. **EZRA, Sandzer-Bell.** *Astromusik.* s.l. : Sync Book Press, 2014. ISBN 978-06-920-2266-5.
8. **JEFF, Maysh.** *THE CODE: A declasssified and unbelievable hostage rescue story.* s.l. : The Verge, 2015.
9. **JURAJ, Pospíšil.** *Hudobná teória pre konzervatóriá.* Bratislava : Slovenské pedagogické nakladateľstvo, 1985.
10. **ANTÓNIA, Droppová.** *Elementárna hudobná teória.* Prešov : Prešovská univerzita, 1998. ISBN 80-88697-39-5.
11. **CATHERINE, Schmidt-Jones.** *The Basic Elements of Music.* ISBN 978-1-312-48694-2.
12. **MasterClass. Music 101:** What Is Harmony and How Is It Used in Music? [Online] 2. 7 2019. https://www.masterclass.com/articles/music-101-what-is-harmony-and-how-is-it-used-in-music#what-is-harmony.
13. **—. Songwriting 101:** Learn Common Song Structures. [Online] 22. 10 2019. https://www.masterclass.com/articles/songwriting-101-learn-common-song-structures#what-is-song-structure.
14. **Wikipedia, Thirty-two-bar form** -. [Online] https://en.wikipedia.org/wiki/Thirty-two-bar_form.
15. **DAVID, Coelle.** *The Complete Guide to JFugue: Programming Music in Java.* 2008.
16. **AUGUSTE, Kerckhoffs.** *La Cryptographie Militaire.* 1883.
17. **Wikipedia, Confusion-and-diffusion** -. [Online] https://en.wikipedia.org/wiki/Confusion_and_diffusion.

18. **JOAN, Daemen -- VINCENT, Rijmen.** *AES submission document on Rijndael.* 1998.
19. **KELLY, Sloan.** *Finite State Machines.* 2019. 10.1007/978-1-4842-4533-0_19..
20. **JAMIE, Henke.** *Basic Concepts of Music Theory.* s.l. : University of Wisconsin-Madison.
21. **MuseScore 3 - Príručka.** [Online] https://musescore.org/sk/pr%C3%ADru%C4%8Dka.
22. [Online] https://www.harm.cz/other/hf.pdf.
23. [Online] https://github.com/xsobotap/Music-cipher.

Príloha

Súčasťou bakalárskej práce je .zip súbor, v ktorom je spomenutá Java implementácia kryptosystému. Súbor Music_Cipher obsahuje dokument **Architektúra systému Musicpher** vysvetľujúci, ktoré triedy a moduly sú zodpovedné za ktorú operáciu. Taktiež obsahuje príklad použitia systému a opis errorov, ktoré sa môžu vyskytnúť.