UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

# HĽADANIE PODOBNOSTÍ V MELÓDIÁCH ĽUDOVÝCH PIESNÍ

**Diplomová práca**

**2020** **Bc. Michal Mižák**

**UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH**
**PRÍRODOVEDECKÁ FAKULTA**

# HĽADANIE PODOBNOSTÍ V MELÓDIÁCH ĽUDOVÝCH PIESNÍ

**Diplomová práca**

| | |
|---|---|
| Študijný program: | Informatika |
| Študijný odbor: | 9.2.1. Informatika |
| Školiace pracovisko: | Ústav informatiky |
| Vedúci práce: | doc. RNDr. Stanislav Krajči, PhD. |

**Košice 2020** **Bc. Michal Mižák**

Univerzita P. J. Šafárika v Košiciach
Prírodovedecká fakulta

# ZADANIE ZÁVEREČNEJ PRÁCE

**Meno a priezvisko študenta:** Bc. Michal Mižák
**Študijný program:** Informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
**Študijný odbor:** Informatika
**Typ záverečnej práce:** Diplomová práca
**Jazyk záverečnej práce:** slovenský
**Sekundárny jazyk:** anglický

**Názov:** Hľadanie podobností v melódiách ľudových piesní

**Názov EN:** Looking for similarities in tunes of folk songs

**Cieľ:**
1. Rozšíriť algoritmus vytvorený v bakalárskej práci "Hľadanie podobností v textoch ľudových piesní" o porovnávanie melódií
2. Implementovať príslušné etnomuzikologické algoritmy
3. Spracovať a vizualizovať výsledky algoritmov

**Literatúra:**
[1] Lubomír Tyllner, Ondřej Skovajsa, Hana Vaňková (eds.). 2017. K otázkám typologie tradiční hudby. Praha
[2] Peter Knees and Markus Schedl. 2016. Music Similarity and Retrieval: An Introduction to Audio- and Web-Based Strategies (1st ed.). Springer Publishing Company, Incorporated.
[3] David Meredith. 2015. Computational Music Analysis (1st ed.). Springer Publishing Company, Incorporated.

**Vedúci:** doc. RNDr. Stanislav Krajči, PhD.
**Oponent:** RNDr. Ondrej Krídlo, PhD.
**Ústav :** ÚINF - Ústav informatiky
**Riaditeľ ústavu:** RNDr. Ondrej Krídlo, PhD.

**Dátum schválenia:** 31.03.2020

## Poďakovanie

Ďakujem vedúcemu diplomovej práce doc. RNDr. Stanislavovi Krajčimu, PhD. za pripomienky, tvorivé nápady a riešenia problémov pri tvorbe tejto práce. Veľká vďaka patrí aj všetkým, čo ma nasmerovali pri formovaní cieľov práce a tým, čo mi poskytli cenné digitalizované zdroje použité v tejto práci.

## Abstrakt

Táto práca sa zaoberá základnou ideou, postupmi a algoritmami hľadania podobností v melódiách ľudových piesní. Postupne popisujeme potrebné teoretické základy z muzikologickej a etnomuzikologickej vednej oblasti. Ďalej tieto postupy skúmame na základe poznatkov z dvoch hlavných vedeckých oblastí: Získavanie hudobných informácií (*Music information retrieval*) a Výskum folklórnych piesní (*Folk song research*). Popisujeme a snažíme sa nájsť prístup, ktorým môžeme tieto metódy čo najlepšie uspôsobiť na porovnávanie slovenských ľudových piesní. Popisujeme problémy, s ktorými sme sa pri implementácii stretli a nakoniec analyzujeme výsledky navrhnutého finálneho algoritmu. Práca nadväzuje na našu bakalársku prácu *Hľadanie podobností v textoch ľudových piesní.*



**Kľúčové slová:** *folklór, ľudové piesne, porovnávanie melódií, hľadanie podobností, melodické rodiny, model vektorového priestoru..*

## Abstract

The thesis focuses on the main idea, processes and algorithms of discovering similarities in folk song melodies. Necessary theoretical knowledge in musicological and ethnomusicological science are successively described. The processes are subsequently investigated using knowledge of two main scientific branches: Music information retrieval and Folk song research. An approach how to utilize the two methods to compare Slovak folk songs is described and searched for. The problems encountered during the implementation are described and results of proposed final algorithm are analysed. The thesis is a follow-up to our earlier bachelor thesis *Discovering similarities in folk song lyrics*.

**Keywords:** *folk art, folk songs, comparison of melodies, discovering similarities, tune families, vector space model.*.

# Obsah

# Úvod

Podobne ako pri bakalárskej práci *Hľadanie podobností v textoch ľudových piesní*, bola aj pre túto prácu inšpiráciou hudba a folklór. Pri vypracovávaní bakalárskej práce sme si uvedomili, že prístup, ktorý sme navrhli na porovnávanie textov, je navrhnutý veľmi vhodne. Vhodne v tom zmysle, že jeho veľká časť sa dá využiť pri porovnávaní iných objektov nielen textov. Najzaujímavejšia bola pre nás myšlienka porovnávania melódií piesní.

Slovensko je krajina bohatá na mnoho vecí a jednou z jej najväčších pokladov je jej folklór. Slováci sú veselý národ a prácu, oslavy a zábavu v minulosti vždy doprevádzali spev, hudba a radosť z maličkostí. Piesne sa tradovali z pokolenia na pokolenie, od spevákov k spevákom a od muzikantov k muzikantom. V istom momente začali byť ľudové piesne zapisované do zbierok odborníkmi a študovanými umelcami z rôznych kútov Európy. Zapisoval sa najmä text, ale miestami sa zapisovala aj melódia.

Teraz nastal čas digitalizácie týchto zápisov a úlohou vedcov je tieto digitalizované dáta analyzovať a skúmať. V tejto práci sme sa sústredili na fakt, že piesne prešli historickým, geografickým a kultúrnym vývojom, čo malo za následok ich textovú, ale aj melodickú modifikáciu. Zaoberať sa budeme práve melodickou modifikáciou.

Povedať, či pri dvoch rôznych melódiách ide o variant jednej piesne je netriviálna úloha a často sa na nej nezhodujú ani odborníci v muzikovedných oblastiach. Naším cieľom je využiť exaktné algoritmy, ktoré budú piesne porovnávať osobitne pre každý starostlivo vybraný atribút piesne. Každú vlastnosť bude skúmať rôzna konfigurácia algoritmu a nakoniec budeme tieto výsledky agregovať do jednej aproximácie percentuálnej podobnosti medzi piesňami. Táto agregácia má formu grafu, kde uzly reprezentujú piesne a ohodnotené hrany medzi nimi ich podobnosť. Nakoniec sa algoritmami na hľadanie komunít v grafe pokúsime rozhodnúť, ktoré piesne pochádzajú z jednej melodickej rodiny – skupiny zápisov piesní, ktoré sú si navzájom variantmi.

Zmysel takéhoto systému vidíme vo všeobecnom zjednodušení práce etnomuzikológov, výskum migrácie piesní po krajine (toto je možné len ak databáza obsahuje aj informáciu o regióne, z ktorého pochádza pieseň), zlepšovanie webových sídiel mapujúcich digitalizované dáta a ako bonus aj istú formu poloautomatizovanej korekcie

nesprávnych zápisov v zborníkoch.

V prvej kapitole zadefinujeme základné pojmy hudobnej teórie nutné pre pochopenie tejto práce. V druhej zadefinujeme pojmy problematiky variácie piesní a ilustrujeme si ju na príkladoch. V tretej kapitole zanalyzujeme spôsoby zaznamenávania piesní a databázu, ktorú máme k dispozícii. Vo štvrtej sa pozrieme na existujúce riešenia analýzy podobnosti piesní. V piatej v krátkosti charakterizujeme technológie, s ktorými budeme pracovať v šiestej a siedmej kapitole, ktoré sa zaoberajú návrhom, implementáciou a analýzou algoritmu. V ôsmej kapitole ukážeme zaujímavé výsledky algoritmu.

# 1 Úvod do nutnej hudobnej teórie

## 1.1 Tón

Tón je zvuk istej výšky (angl. *pitch*). Výška tónu sa meria v hertzoch a je priamo úmerná jeho kmitočtu. Pod vysokým (hovorovo tenkým) tónom si vieme predstaviť napríklad štrngnutie vínových pohárov, piskot hlodavcov alebo detský hlas. Naopak, nízky (hovorovo hlboký) tón je napríklad trúbenie lode alebo zvuk tympánu.

## 1.2 Doba

Doba je základný pravidelný pulz skladby. Predstavuje pevne danú časovú jednotku, na základe ktorej sa určuje chronológia udalostí v skladbe (ktoré tóny majú kedy a ako dlho znieť). Podľa Medzinárodnej sústavy jednotiek *SI* používame ako základnú jednotku času jednu sekundu. Trvanie sekundy je pevné, nemenné. Doba je však abstraktnejší pojem, keďže v rámci jednej hudobnej skladby sa trvanie jednej doby môže dynamicky meniť pri zrýchľovaní alebo spomaľovaní skladby. Po zrýchlení skladby jedna doba trvá kratšie, po spomalení dlhšie.

## 1.3 Tempo

Pre špecifikáciu toho, kedy má v piesni zaznieť ktorý tón, nám doba postačuje. Potrebujeme však aj niečo, čo namapuje trvanie jednej doby na pevnú jednotku trvania času. Na to slúži hodnota *BPM* (anglicky *beats per minute*), ktorá symbolizuje počet dôb na jednu minútu. V minulosti sa používali aj voľnejšie pojmy ako *Allegretto*, *Moderato* alebo *Largo*, ktoré (väčšinou) určujú tempo len rámcovo a spoliehajú sa na hudobníkov alebo dirigentov, aby vnútorne sami určili vhodné *BPM*, prípadne, aby ho určili na základe obdobia, v ktorom skladba bola napísaná alebo iných parametrov skladby.

## 1.4 Takt

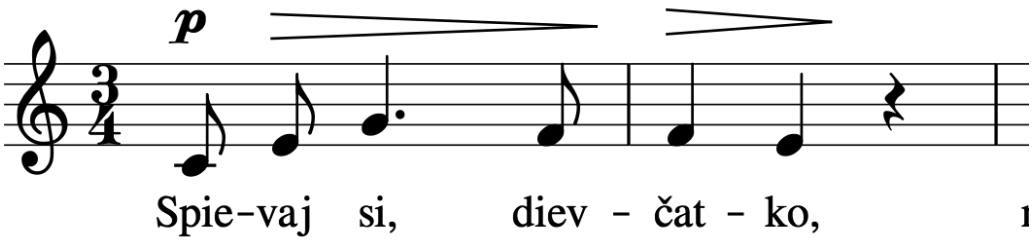
Takt je časový úsek, ktorý korešponduje dopredu špecifikovanému počtu dôb. Pre jednoduchosť si takt vieme predstaviť tak, že hudobnú skladbu rozdelíme na niekoľko rovnako dlhých častí (podľa počtu dôb) a takt je práve jedna z týchto častí (pri výučbe na základných školách sa niekedy využíva metafora domčeka pre noty). Počas skladby sa môže počet dôb v takte zmeniť. Tento jav sa, aj keď zriedkavo, vyskytuje aj v ľudovej piesni. Predstaviť si to vieme ako pieseň, kde sa sloha spieva ako valčík (tri doby v takte) a refrén ako polka (dve doby v takte).

## 1.5 Melódia

Melódia je sekvencia za sebou idúcich tónov. Tieto tóny môžu byť rôznej dĺžky. Môžu nasledovať bezprostredne za sebou alebo môžu medzi nimi byť prestávky, kedy neznie žiaden tón. Melódia týmto nesúvisí len s výškou, ale aj s dĺžkou tónov a prestávok medzi nimi. Každá ľudová pieseň má svoju melódiu[1].

## 1.6 Rytmus

Etymológia slova rytmus pochádza z gréckeho slova *rhythmos*, ktoré znamená plynutie a je odvodené od slova *rhein*, čo znamená plynúť, tiecť. Pre naše potreby budeme slovo rytmus používať zjednodušene, a to na popísanie rytmu melódie - sekvencie dĺžok tónov a prestávok medzi nimi. Dĺžka je reprezentovaná násobkom jednej doby[2].




Obr. 1: Útržok piesne

Na príklade vyššie vidíme útržok ľudovej piesne „*Spievaj si, dievčatko*“. Jednotlivé tóny sú graficky reprezentované notami[3]. Rôzny tvar nôt reprezentuje ich dĺžku

[1] V iných, exotickejších kultúrach, je definícia niekedy náročnejšia, na Slovensku to však platí vždy.

[2] Bežne ide len o súčet celého čísla a jednej polovice alebo jednej tretiny

[3] Aj keď sa snažíme text formulovať hudobne čo najjednoduchšie, kvôli lepšiemu porozumeniu práce odporúčame čitateľovi naučiť sa základy čítania nôt z notovej osnovy.

a to, ako je nota v rámci notovej osnovy vysoko (na ktorej, resp. medzi ktorými čiarami sa nachádza) ukazuje výšku tónu. Melódia je sekvencia týchto tónov, v našom prípade spievaná fráza *Spie-vaj-si-diev-čat-ko* má každú slabiku prepojenú s nejakým tónom. Ako je vidieť, tieto tóny sú rôznej výšky až na dve slabiky *diev-čat*, ktoré sú rôzne len dĺžkou – rytmicky. *Rytmus melódie* je sekvencia dĺžok týchto nôt korešpondujúcim jednotlivým slabikám. Prvý *takt* je zľava ohraničený zaznačením trojštvrťového taktu a končí prvou vertikálnou čiarou. Druhý takt je oddelený dvomi vertikálnymi čiarami.

## 1.7 Úvod do temperovaného ladenia

Ladenie je veľmi komplikovaná oblasť hudobnej teórie. Popisuje, na akej frekvencii majú znieť jednotlivé tóny. Pre potreby našej práce stačí poznať jediné ladenie – temperované, ktoré je v západných krajinách najrozšírenejšie. Je dôležité povedať, že rôzne, pre nás exotické, ladenia, ako napríklad rôzne mikrotonálne indické stupnice, poznajú desiatky rozličných tónov, čo otvára veľmi veľa možností pri hudobnej tvorbe. Nesie to však so sebou rôzne komplikácie a aj preto sa v minulosti spoločnosť zhodla na kompromise a zvolila taký, čo najjednoduchší systém ladenia, ktorý dostatočne aproximuje absolútne (perfektné, dokonalé) ladenie. Systém sa používa dodnes. Existuje v ňom 12 rôznych tónov, ktoré sa cyklicky opakujú. Máme 7 fundamentálnych tónov – fundamentálnymi ich nazývame len vďaka ich jednoduchšiemu názvu, inak sú si všetky tóny rovnocenné. Sú to práve všetky biele klávesy na klavíri: *C D E F G A H*.

Anglicky sa tón *H* nazýva *B*. Spomíname to preto, pretože potom sa na tieto názvy dá pozrieť ako na cyklické posunutie prvých 7 písmen abecedy:

*C D E F G A H*

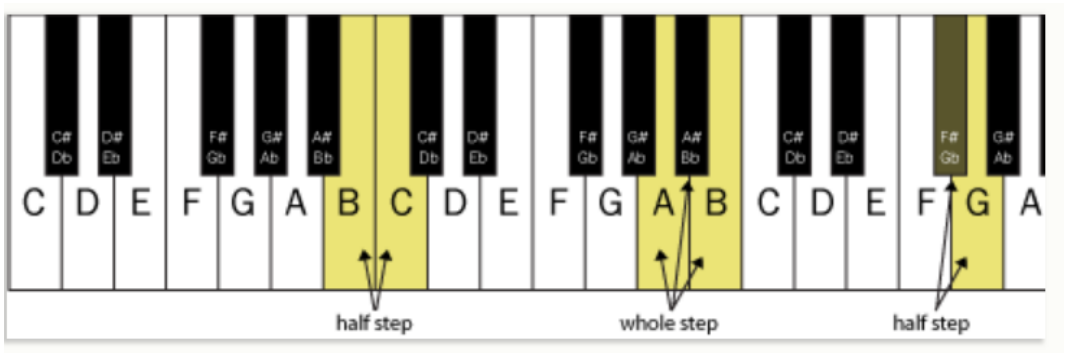
zmena názvu =>

*C D E F G A B*

cyklické posunutie =>

*A B C D E F G*.

Podľa názvov, ktoré sa používajú na Slovensku, prípona *-is* naznačuje zvýšenie o poltón a prípona *-es* zníženie o poltón. Poltón v tomto ladení znamená najnižšiu možnú vzdialenosť medzi dvoma tónmi. Graficky si poltón vieme načrtnúť pomocou klaviatúry:




Obr. 2: Klaviatúra[4]

Na obrázku vidíme časť klaviatúry, na ktorej sú žltou farbou označené poltóny, alebo anglicky *half step* = polovičný stupeň / krok. Je vidieť, že poltón sa nachádza medzi každou dvojicou susedných klávesov. Dva biele (resp. podľa obrázku aj žlté) klávesy v tomto prípade nazveme susedné len vtedy, keď medzi nimi nie je čierny kláves. V konečnom dôsledku existuje 12 tónov, na obrázku je zaužívané anglické názvoslovie, podľa slovenskej nomenklatúry to je napríklad[5]:

***C Cis D Dis E F Fis G Gis A Ais H***.

Všetky zvýšené tóny s príponou *-is* by sme vedeli pomenovať podobným spôsobom pomocou znižovania pomocou *-es*, napríklad ***Cis*** znie v temperovanom ladení rovnako ako ***Des***. V prípade ***F*** to je zaujímavejšie, pretože ***F*** je to isté ako ***Eis***, keďže ***E*** a ***F*** sú od seba vzdialené len poltón.

## 1.8 Oktáva

Zastavíme sa ešte pri pojme oktáva. Vyššie sme spomínali, že temperované ladenie sa skladá z 12 cyklicky sa opakujúcich tónov. V praxi to znamená, že tóny vieme zoradiť lineárne od najnižšieho po najvyšší nasledovne:

***C Cis D Dis E F Fis G Gis A Ais H C Cis D Dis E F Fis G Gis A Ais H C***.

Tón ***C*** sa nachádza v reťazci trikrát. Oktávou nazveme index cyklu, v ktorom sa nachádza (alebo koľkýkrát sme ho už pri čítaní zľava videli). Platí, že čím vyššia je oktáva tónu, tým je vyššia frekvencia tónu. Čiže prvé ***C*** nazveme o oktávu nižšie ako druhé ***C*** a to o oktávu nižšie od tretieho ***C***.

Sekvenciu za sebou idúcich tónov s rozdielom jedného poltónu nazývame *chromatickou stupnicou*.

[4]Zdroj obrázku: `http://musictheoryfundamTentals.com/MusicTheory/intervals_part1.php`

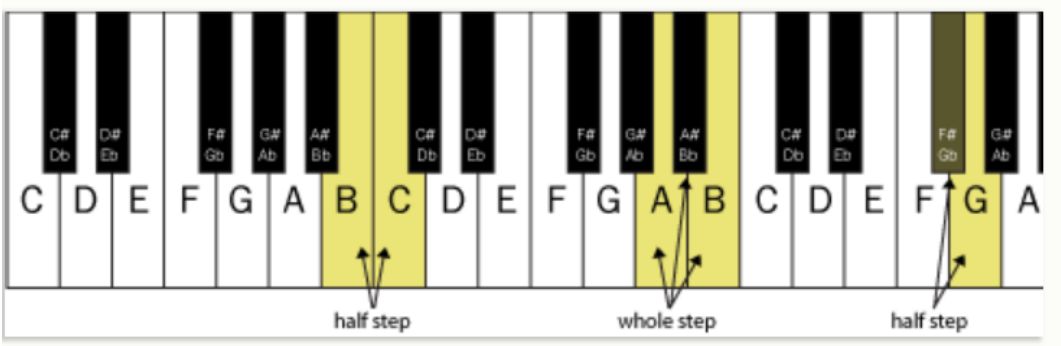
[5] Dalo by sa to zapísať mnohými spôsobmi kvôli enharmonickej zámene tónov.

## 1.9 Stupnice

Stupnica je ľubovoľná množina nôt zoradených podľa svojej výšky. Teraz vieme, že tieto noty reprezentujú tóny z našej množiny 12 tónov. Stupnica je teda ľubovoľná usporiadaná podmnožina tejto množiny. Najčastejšie sa budeme stretávať s durovými stupnicami. Jedna z najbežnejších durových stupníc je napríklad *C dur*:

*C D E F G A H*.

Ďalej stupnica pokračuje cyklicky, znova od *C*, podobne ako pri predošlom príklade k oktáve, len so 7 tónmi namiesto 12.




Obr. 3: Klaviatúra

Keď sa znova pozrieme na predošlý obrázok, uvidíme, že medzi *C* a *D* sú 2 poltóny, medzi *D* a *E* tiež, medzi *E* a *F* jeden atď. Dá nám to sekvenciu intervalových skokov

2-2-1-2-2-2-1.

Ak pridáme do sekvencie aj tóny, dostaneme

*C*-2-*D*-2-*E*-1-*F*-2-*G*-2-*A*-2-*H*-1-*C*.

Pomocou numerickej sekvencie intervalových skokov (bez tónov) sa teda „vyšplháme“ od tónu *C* späť na tón *C*, len o oktávu vyššie. Keď začneme od tónu *D* namiesto tónu *C*, tak dostaneme *D dur*:

*D E Fis G A H Cis*

a znova *D*. Po spojení sekvencií intervalov a tónov dostaneme podobne ako pri *C dur*

*D*-2-*E*-2-*Fis*-1-*G*-2-*A*-2-*H*-2-*Cis*-1-*D*.

Je vidieť, že táto sekvencia intervalových skokov ostala rovnaká. Druhá najčastejšia stupnica je molová, napríklad stupnica ***a mol***:

***A H C D E F G A***.

Molová a durová sekvencia intervalových skokov sú len navzájom cyklicky posunuté postupnosti, pretože ak z durovej sekvencie vezmeme 2 a 1 z konca a „nalepíme“ ich na začiatok, dostaneme zo sekvencie

2-2-1-2-2-2-1

sekvenciu

2-1-2-2-1-2-2

po pridaní tónov

***A***-2-***H***-1-***C***-2-***D***-2-***E***-1-***F***-2-***G***-2-***A***.

Takto vieme pomocou 7 rôznych cyklických posunutí jednej intervalovej sekvencie dostať 7 rôznych stupníc nazývaných módmi. Zo všetkých módov (jónsky, dórsky, frygický, lydický, mixolydický, aiolský a lokrický) je okrem jónskeho (durová stupnica) a aiolského (molová stupnica) najčastejší mód lydický, ale častý je aj dórsky a mixolydický. Každý z týchto názvov reprezentuje cyklické posunutie sekvencie intervalových skokov durovej stupnice a týmto, ak zanedbáme posunutie o oktávu, tak všetky módy vieme vyskladať z jednej množiny 7 tónov. Napríklad zo stupnice ***C dur*** vieme vytvoriť lydický mód tak, že vezmeme prvé 3 tóny z ***C dur*** a „nalepíme“ ich na koniec postupnosti (a posunieme o oktávu vyššie) nasledovne:

***C D E F G A H***

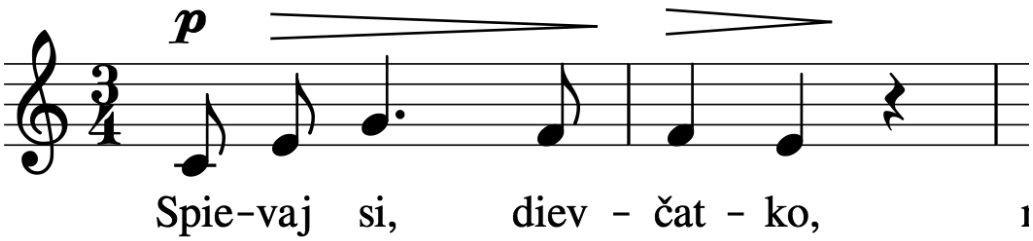
zmeníme na

***F G A H C D E***

Takýchto množín je dohromady 12 kvôli tomu, že naše ladenie má práve 12 tónov a z toho 12 rôznych durových stupníc. Poznáme mnoho ďalších tónových štruktúr (napr. v archaickejších piesňach veľmi časté tónové rady), kvôli jednoduchosti ich však spomínať v texte nebudeme.

## 1.10 Tónina a modulácia

Tónina pomenúva vzťah tónov hudobnej skladby k určitej stupnici. Ak sa v skladbe vyskytujú najmä tóny zo stupnice `G dur`, v kontexte danej skladby správnejšie hovoríme o tónine, nie o stupnici `G dur` (aj keď hovorovo tieto dva pojmy splývajú). Tento pojem potrebujeme kvôli javu, ktorý sa nazýva *modulácia piesní*. Už sme hovorili o tom, že pomocou intervalovej sekvencie vieme vybudovať rôzne stupnice na základe toho, od akého tónu začneme. Ak sa pozrieme na stupnicu `D dur` a na stupnicu `C dur`, je vidieť, že keď všetky tóny z `D dur` znížime o 2 poltóny, dostaneme `C dur`. Je to prirodzené, pretože na vytvorenie oboch stupníc sme použili rovnakú intervalovú sekvenciu, ale rôzne počiatočné tóny, ktoré sú vzdialené práve o 2 poltóny. Posúvanie sekvencií tónov alebo, kompaktnejšie povedané, melódií, o fixný počet poltónov nahor alebo nadol nazývame *modulácia*. Hovoríme teda o operácii, ktorú vieme aplikovať na ľubovoľnú melódiu.




Obr. 4: Útržok piesne

Pozrime sa znova na pieseň `Spievaj si, dievčatko`. Melódia sa skladá z nôt, ktoré postupne reprezentujú tóny

`C E G F F E`

Intervalová sekvencia tejto melódie je už ťažšie popísateľná, pretože v rámci melódie sa aj klesá. Zavedieme teda notáciu $+i$ a $-i$, kde $i$ je prirodzené číslo a $+$ naznačuje stúpanie o $i$ poltónov, - klesanie. Z tejto piesne dostaneme teda

```
+4 +3 -2 +0 -1
```

Touto notáciou dokážeme reprezentovať *relatívne vzťahy tónov*. To znamená, že keď ako počiatočný tón zvolíme iný tón, napríklad tón `D`, dostaneme melódiu

`D Fis A G G Fis`,

o ktorej hovoríme, že je *zmodulovaná* o 2 poltóny (alebo 1 celý tón) vyššie od pôvodnej. Táto melódia bežnému uchu znie rovnako, ak človek nemá veľmi zriedkavú schopnosť

absolútneho sluchu. Ak si bežný poslucháč pustí tieto dve melódie za sebou s dostatočným časovým odstupom, nezistí, že je to v skutočnosti melódia, ktorá s predošlou nemá spoločný ani jeden tón. Zachováva však všetky *relatívne vzťahy*. Modulácia je bežnou praxou pri spievaní piesní, pretože každý interpret spieva pieseň v rámci svojich rozsahových možností – ženy a deti majú prirodzene vysoko položené hlasy a vďaka tomu spievajú vysoko a, naopak, muži spievajú v nižších polohách. Rozdiel v preferovaných tóninách môže byť od speváka k spevákovi obrovský.

# 2 Základné pojmy a ilustračný príklad

## 2.1 Neexistencia dokonalého algoritmu na porovnávanie piesní

V tejto sekcii poukážeme na obrovský rozsah zložitosti problému podobnosti melódií.

### 2.1.1 Autorské práva

Komerčne najdôležitejšia oblasť, kde sa skúma podobnosť piesní, je oblasť vymáhania nárokov na autorské práva melódií. Pri vidine vymáhania peňazí je podávané veľké množstvo žalôb na tvorcov kvôli podozreniam na okopírovanie piesne alebo jej úryvku od iných tvorcov. To, že sa to týka aj veľkých umelcov, potvrdzujú žaloby piesní ako *Led Zeppelin – Stairway to Heaven*, *Katy Perry – Dark horse*, *Ed Sheeran – Thinking out loud* (bola žalovaná kvôli podobnosti k piesni *Marvin Gaye – Let's get it on*). Žaloba *Thinking out loud* sa opiera okrem iného aj o harmonickú podobnosť akordickej postupnosti, ktorú využíval už *J. S. Bach* v jeho diele *Lobt Gott, ihr Christe, allzugleich*. Absurdnosť tejto žaloby názorne vysvetľujú vo svojich videách napríklad jazzoví muzikanti a YouTube tvorcovia Adam Neely a Rick Beato.

Žaloby prepájajú muzikológiu s právom, čo vedie k mnohým absurdnostiam. Jednou z nich je počin právnika a hudobníka Damiena Riehla, ktorý sa so svojím kamarátom programátorom Noahom Rubinom postaral o vygenerovanie všetkých existujúcich melódií. Týmto experimentom chceli poukázať na absurdnosť celého právneho základu vymáhania autorských práv melódií. Hovoria o akomsi „mínovom poli“, po ktorom kráča každý tvorca nových melódií, pretože existuje konečný počet melódií, ktoré môžu existovať. Skladateľ a hudobník Oli Frekena to vyčíslil na zhruba 75 miliárd melódií dĺžky 10 a ak vezmeme do úvahy aj rytmus, dostaneme približne $8.25 \cdot 10^{19}$ melódií[1].

[1] Zdroj: `https://plus.maths.org/content/how-many-melodies-are-there`

To v praxi znamená, že s každou novou melódiou sa zmenšuje počet originálnych melódií, ktoré nemôžu byť žalované. Z právneho hľadiska uložením na médium (pevný disk) Damien dosiahol to, že môže žalovať ľubovoľnú skladbu, pretože každá využije najmenej jednu z týchto umelých melódií. Žaloby by skrátka podporil tým, že pieseň vznikla neskôr ako tieto generované dáta a keďže tieto piesne zverejnil na svojej webstránke, existuje pravdepodobnosť, že ich žalovaný tvorca počul.

Tieto extrémne prípady sa týkajú umelej tvorby a je zrejmé, že ak ide o problémy s autorskými právami, automatizovaný systém na porovnávanie melódií nemá veľký zmysel. Analýza dvoch piesní nie je tak časovo náročná ako výskum veľkého datasetu, preto je vždy výhodnejšie, keď je vykonaná muzikológmi. Takto bude zároveň aj hĺbkovejšia.

Faktom však ostáva, že autori svoju žalobu podporovali poukázaním na veľmi vzdialené, až priam algoritmicky neuchopiteľné podobnosti. Podobnosť autorských (západných) melódií, v tomto prípade celých piesní, sa aj historicky javí ako veľmi subjektívna a stačí rovnaký *feel* a farba zvukov, aby bežnému človeku pieseň pripomínala inú pieseň. Nehovoriac o tom, že harmonický podklad popových piesní je až schematicky rovnaký naprieč celým spektrom vrcholných hudobných rebríčkov.

### 2.1.2 Melodické rodiny

V zahraničnej literatúre je pojem melodická rodina (anglicky *tune family*) definovaný rôzne. V práci sa budeme prikláňať k tejto definícii, ktorá zhŕňa viacero rôznych definícií:

*Rodina melódií (alebo melodická rodina) je skupina melódií prepojených melodickou korešpondenciou (súhlasom), presnejšie vo všeobecnej melodickej kontúre, dôležitých intervaloch a promimentných akcentovaných tónoch. V rámci jednej rodiny sa môžu vyskytovať rozdiely v rytmických vzoroch, módoch a textoch. Takéto melódie sa vyvinuli z jednej koreňovej melódie, ktorá bola pozmenená procesom variácie a imitácie, keď bola tradovaná ústnym podaním.*

Pojem budeme spájať s blízko prepojeným konceptom *vandrujúcich melódií*, ktorý pomenúva jav podobných melódií v rôznych geografických oblastiach. Tento veľmi zaujímavý koncept vieme uviesť na príklade zemplínskych melódií, ktoré boli prinesené po obchodných trasách, ktoré prechádzali rôznymi kútmi Zemplína. Aj to mohlo mať za následok vplyv na folklórne bohaté dediny Veľké Zalužice (neskôr sa spojila s Malými Zalužicami a spolu teraz tvoria Zalužice), Sliepkovce či Budkovce, v ktorých nájdeme veľmi exotické melódie pre túto oblasť. Týmito cestami sa dalo dostať do Maďarska, Rumunska, Srbska či Poľska. To v konečnom dôsledku prinieslo z a do tejto oblasti veľa zaujímavých tanečných a speváckych melódií. Fenomén vandrovania piesní potvrdzuje

spievanie maďarských piesní (aj bez melodickej modifikácie) so slovenskými textami a opačne. Príkladom je zemplínska pieseň *U Rybnici* (spievajú sa aj iné dediny, ako tomu je zvykom v ľudových piesňach) *na valaľe*. Tejto téme sa v minulosti venovalo mnoho odborníkov (napríklad v [4]) a skúmali, kde piesne skutočne vznikli – či sú maďarské alebo slovenské.

Pojem melodickej rodiny budeme v práci chápať aj tak, že piesne, ktoré sú v jednej melodickej rodine, sú navzájom *variantmi*.

### 2.1.3 Slovenské ľudové piesne

Pomôžeme si citáciou manželov Elschekovcov z [1] (str. 59):

*„Z doterajšieho rozboru hudobnoštrukturálneho ústrojenstva ľudovej piesne je zrejmé, že ľudová pieseň je organizmom veľmi zložitým, mnohovrstvovým; nie je útvarom ani jednoduchým, naivným, či primitívnym, ako sa to často tvrdí. Preniknúť do zložitého ústrojenstva ľudového hudobného myslenia je občas ťažšie, ako urobiť prehľadný rozbor skladieb umelej hudby, v ktorých je niekedy viac šablónovitosti, konvenčnosti a schematizmu ako v neviazanom útvare hudobného folklóru.*“

Tento citát implikuje, že nájsť pravidelnosť v ľudovej piesni je náročné, niekedy dokonca náročnejšie, ako v umelej hudbe. Práve kvôli tomu sa problematike podobnosti venuje mnoho odborníkov a dosahujú významné výsledky v oblasti nachádzania vzorov v ľudových piesňach. Tieto vzory sú ale často špecifikované len pre podskupiny piesní (či už historicky, geograficky alebo typovo – tanečné, parlandovité...). V praxi to pre nás znamená to, že naším cieľom nesmie byť algoritmus, ktorý otestuje určitú vlastnosť piesní a spoľahlivo povie, či každé dve piesne sú alebo nie sú navzájom variantmi – takýto **neexistuje**.

Z tohto dôvodu bude náš algoritmus testovať viac rôznych vlastností melódií, ktoré len môžu naznačovať to, že piesne sú variantmi. Môže sa stať, že na základe jednej z týchto vlastností budú piesne totožné a na základe inej budú úplne rôzne – budú mať nulovú podobnosť. Na konci budú tieto osobitné výsledky agregované do jednej aproximácie podobnosti.

Náš systém nikdy nebude stopercentne spoľahlivý a vždy bude potrebovať dodatočnú kontrolu odborníkom. Samostatne môže slúžiť len ako aproximačný systém v prípade, že nám nezáleží na dokonalej korektnosti výsledkov – napríklad prehľadová podobnosť pre používateľa na portáloch ako web projektu *Ludo slovenský*. Druhý spôsob využitia je ako systém, ktorý bude odporúčať odborníkom dvojice piesní, ktoré sú potenciálnymi variantmi, respektíve množiny piesní, ktoré sú kandidátmi na melodickú rodinu. Potvrdiť, či sú aj reálne variantmi, musia títo odborníci a ak áno, tak budú môcť výsledok popísať a skúsiť vydedukovať historickú, geografickú alebo inú podstatu

tohto variantu na základe dát zo zbierok, v ktorých boli tieto piesne zaznamenané. Je dôležité poznamenať, že je to takto nutné aj kvôli tomu, že takmer žiadne dáta zo zbierok slovenských piesní nie sú (verejne) dostupné v elektronickej forme použiteľnej programátorsky (existujú pdf skeny, tie však využiť nevieme).

Modelový prípad by bol napríklad taký, že náš algoritmus vráti pre dvojicu melódií vysokú podobnosť. Odborníci potom na základe dát zo zborníkov, empirických poznatkov a historických súvislostí identifikujú, že pieseň vznikla a tradovala sa najprv na Zemplíne a potom sa dostala na Horehronie vďaka dievčaťu, ktoré sa tam vydalo.

Náš systém poskytne len hĺbkovú analýzu rôznych vlastností piesní a zhrnutie týchto výsledkov. Takýto uchopiteľný zmysel tomuto výsledku môže kvôli absencií metadát dať zatiaľ len človek a až neskôr budeme môcť uvažovať o využívaní elektronických dát a automatizovane dedukovať príčinu z nich.

## 2.2 Jednoduché prípady variácie

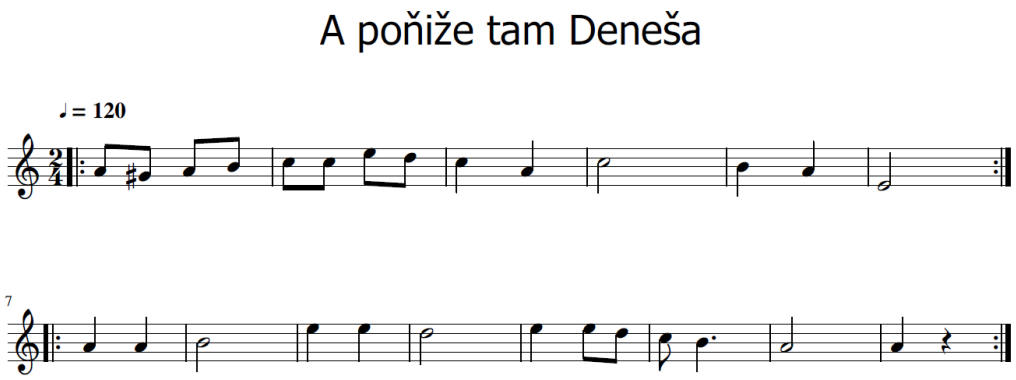
Popíšeme dva typy variácie, ktoré sme v minulosti pozorovali na ľudových piesňach.

### 2.2.1 Kolísavé tóny

S detekciou problému kolísavých tónov sme sa prvýkrát stretli vo folkloristickej praxi. Problém je rozoberaný aj na akademickej pôde, napríklad v článku [5], kde je ako kolísavý tón popisovaná zväčšená kvarta. V databáze kysuckých piesní, ktorou sa článok zaoberal, je to veľmi častý jav. V tomto článku je popisovaná lydická kvarta – štvrtý tón lydického módu, v lydickej stupnici od tónu *C* to je tón *Fis* namiesto tónu *F*, ktorý stupnica štandardne obsahuje. Hovorí sa o kolísaní v rámci jednej piesne, v iných zdrojoch je však popisovaná lydická/ nelydická kvarta v rámci rôznych zápisov tej istej piesne. Z tohto sa dá dedukovať, že chromatická zmena na kvarte je častý jav v rámci variantov piesní. Tento problém sa nám tiež podarilo vyriešiť.

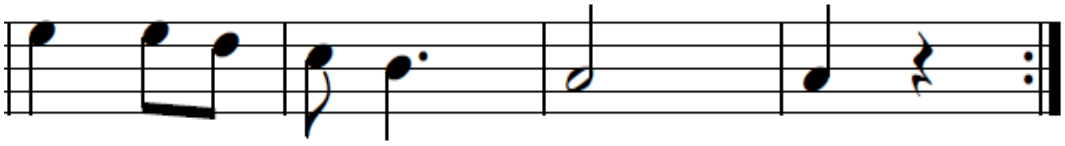
### 2.2.2 Harmonické varianty fráz

Pri prvotnom návrhu bolo cieľom práce navrhnúť algoritmus, ktorý identifikuje aj varianty, ktoré vznikli vokálnou harmonizáciou. Tento jav sme pozorovali vo folkloristickej praxi. Ilustrovať ho môžeme na piesni *A poňiže tam Deneša*.



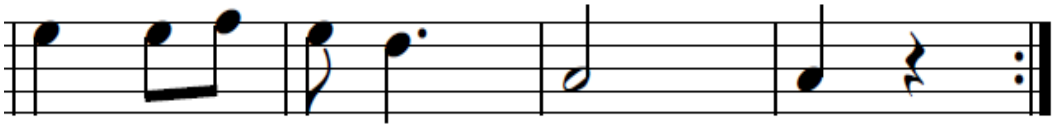

Obr. 5: *A poňiže tam Deneša*

Pozrime sa na záver tejto piesne:



Obr. 6: Harmonizácia melódie *A poňiže tam Deneša*

A na harmonizáciu tejto frázy, ľudovo povedané, horným tercom:



Obr. 7: Koniec melódie *A poňiže tam Deneša*

Tieto dve piesne sú evidentným variantom. Obe tieto frázy sa vyskytujú v ľudových piesňach pomerne často (napríklad aj v známej piesni *Odam śe neodam*, vo fráze *i doma śe pri-dam*) a veľmi často sú navzájom zamieňané kvôli tomu, že jedna je za istých podmienok harmonizácia druhej. Zároveň sú obe melódie príjemné, čo je predpoklad na to, aby si ich speváci spievali samostatne (druhé a tretie hlasy spievané samostatne bez hlavnej melódie často nedávajú zmysel a je teda malá pravdepodobnosť, že si ich speváci budú spievať sólovo). Dá sa teda povedať, že tieto samostatné frázy sú navzájom

harmonickými *variantmi*. Tento pojem sme síce vyššie definovali pre celé piesne, ale dá sa definovať aj v menšej miere – na frázy. Pieseň *A poňiže tam Deneša* a tá istá pieseň končiaca sa harmonizovanou frázou sú teda jednoduchým prototypom toho, ako vznikajú varianty piesní.

Priamočiarym riešením, nad ktorým sme v minulosti uvažovali, je v tomto príklade povedať, že ak sú od seba frázy vzdialené o terciu v rámci stupnice, označíme ich za podobné. To nám však môže dávať nepresnosti napríklad kvôli tomu, že tieto frázy nemusia byť modulované o terciu celé, ale, ako je vidieť aj na príklade, len ich časť (v príklade prvé a posledné dva tóny ostali nezmenené). Videli sme riešenie v prístupe, ktorý povie, že ak sú tóny vzdialené o terciu, tak sú bližšie, ako keby boli vzdialené o sekundu alebo iný interval. Ukázalo sa však, že tento problém je komplikovanejší a dokázali sme ho vyriešiť alternatívne.

## 2.3 Ilustračný príklad

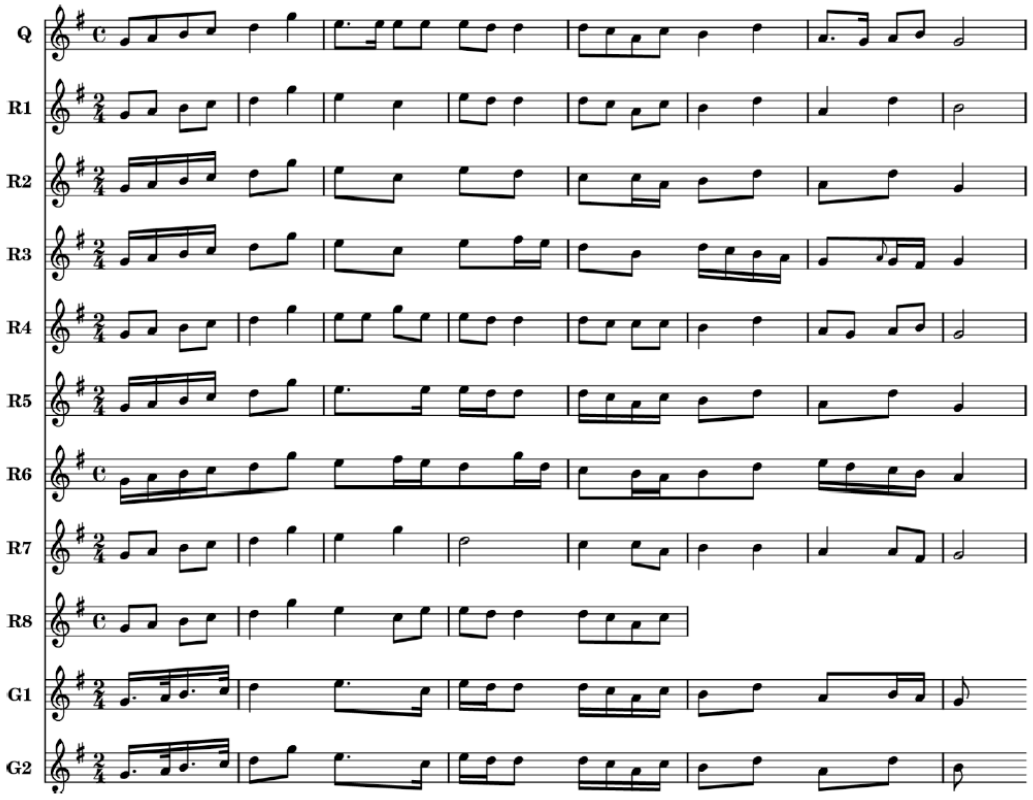



**Figure 2** First lines of the search results. **Q** is the query. **R1** is found by the Danish engine, **R2** and **R3** by the Finnish engine, **R4** and **R5** both by Themefinder and Meldex, **R6** and **R7** by Musipedia, and **R8** by YahMuugle. All melodies are transposed to G major. The titles are: **Q** *Dat gaat naar Den Bosch toe*, **R1** air *Om al Verden er*, **R2** [without title], **R3** *Nelosta*, **R4** *Loot ons noch ens drinken*, **R5** *Ueber die Beschwerden dieses Lebens*, **R6** *Scottisch Simple de Guemene*, **R7** *I'm a little tea pot*, **R8** *Variations* by Aloys Schmitt. Two phrases from the original composition by Pierre Gaveaux, **G1** and **G2**, are added for comparison

Obr. 8: Ukážka podobných melódií z článku [8]

Na hlbšiu ilustráciu podobných piesní využijeme ukážku z práce P. van Kranenburga [8]. Ukážka je vytvorená z piesní, ktoré nie sú slovenské, sú však poväčšine ľudové, a teda charakter ukážky bude veľmi podobný našej problematike.

V tomto článku je využívaný princíp dopyt – výsledky dopytu z databázy. Na obrázku sa nachádza postupne 11 rôznych melódií. V prvom riadku sa nachádza dopytná melódia $Q$, pomocou ktorej boli zadávané konkrétne dopyty do informačných systémov. Postupne rôzne systémy vrátili výsledky dopytu $R1$ – $R8$. Už voľným okom laika je vidieť, že tieto výsledky sú z veľkej časti vizuálne podobné s dopytnou melódiou $Q$ komplikovanejším spôsobom ako predošlé príklady. Napríklad riadok $R7$ je od dopytu $Q$ rytmicky značne rozdielny. Nedá sa však povedať, že medzi nimi neexistuje žiadna

podobnosť – že nemôžu byť navzájom variantmi. Tu by prišlo na rad vyššie spomínané hľadanie príčiny, kontextualizácia piesní a následné potvrdenie, respektíve vyvrátenie, či skutočne ide o variant. Na druhej strane vidíme, že melódia *R7* je v porovnaní s melódiou *R2* značne menej podobná s dopytom $Q$ – je menej pravdepodobné, že sú navzájom variantmi. Výskum bol pre zaujímavosť doplnený ilustratívnymi melódiami *G1*, *G2* z umelej (autorskej, nie ľudovej) hudby.

Funkcia nášho systému bude odlišná od spomínaných informačných systémov (anglicky *retrieval systems*), ktoré slúžia najmä na rýchle vyhľadávanie a intenzívne sa zaoberajú aj efektivitou algoritmov. Náš systém má byť čo najprecíznejší a nezaujíma nás, či bude bežať hodinu alebo deň, lebo výsledok bude statický a nikdy sa vypočítané dáta nebudú meniť. Príklad výsledku nášho algoritmu si môžeme predstaviť tak, že výsledky *R1* až *R8* z dopytu $Q$ bude systém považovať za varianty, či už s vyššou alebo nižšou pravdepodobnosťou. Túto pravdepodobnosť bude reprezentovať percentuálne a zároveň poskytne aj možnosti analyzovať podobnosť hlbšie nad rámec jednoduchých výsledkov dopytu.

# 3 Databáza piesní

## 3.1 Spôsoby zaznamenávania ľudových piesní

Existuje viac spôsobov, akými sa v minulosti zaznamenávali a uchovávali piesne. Zhrnieme tie najrozšírenejšie okrem zrejmého ústneho tradovania.

### 3.1.1 Zápis textov

Prvý, najjednoduchší prístup, je zapisovanie textov a zapamätanie si melódie piesne. Tento prístup bol aj historicky najprístupnejší pre širokú verejnosť. Ľudia, ktorí vedeli písať, si takto zaznamenávali piesne, aby na ne nezabudli. Existujú etnografické výskumy už zo začiatkov 20. storočia, kde starostovia, farári a rôzni odborníci a nadšenci spisovali zvyky a obyčaje. Popritom zaznamenávali aj texty piesní. Spomenieme napríklad zápis svadobného obradu zo zemplínskej obce *Staré*, kde je zaznamenaný celý priebeh svadby aj s rituálmi a obradnými piesňami. Na uchovávanie zvykov a ľudovej reči je tento prístup dostatočný, ale na analýzu melódií je zrejme nedostačujúci. Potvrdzuje to aj fakt, že k textom piesní zo *Starého* nedokázala pred pár rokmi za pomoci výskumných techník priradiť nápevy ani jedna z najvýznamnejších osobností zemplínskeho folklóru – Milan Hvižďák. A tak sú už tieto piesne asi, žiaľ, zabudnuté. Ostal len text.

### 3.1.2 Notácia piesní

Druhý spôsob zaznamenávania piesní je notáciou. Notáciu mohli vykonávať len ľudia s hudobným vzdelaním a tých medzi bežnými ľuďmi nebolo až tak veľa. Našli sa však ľudia ako Béla Bartók, Karol Plicka alebo Ladislav Galko, ktorých zbierky sú ešte stále relevantým zdrojom výskumu.

### 3.1.3 Nahrávky piesní

Najlepším spôsobom, ako piesne uchovávať, je a vždy aj bolo ich nahrávanie. Najlepšie je preto, lebo ľudový spev nie je možné znotovať dokonale. Všetky ozdoby a nuánsy v ľudskom hlase, ktorými speváci obohacujú melódie, sú často špecifické a unikátne pre každú pieseň, pre každú slohu a často dokonca pre každé prevedenie tej istej piesne. Notácia tak, ako ju poznáme, je síce mocný nástroj, ale má svoje limity. Takéto nahrávky robil napríklad *Fonografický a gramofonický archiv České akademie věd a umění*[1] už v dvadsiatych rokoch minulého storočia, keď mobilizoval odborníkov, ktorí vykonávali mnoho terénnych akcií. Tieto nahrávky sú prístupné v rôznych archívoch.

## 3.2 Digitalizácia a zdroj databázy

V našej práci nebudeme využívať nahrávky, ktoré síce majú najlepšiu výpovednú hodnotu o ľudovej piesni, ale získať z nich spoľahlivý symbolický zápis melódie je v momentálnom stave tejto vedeckej oblasti veľmi náročné a v mnohých prípadoch aj prakticky nemožné. Je to spôsobené aj tým, že v súčasných ľudových nahrávkach sa okrem spevu nachádzajú aj iné nástroje, ktoré svojou farbou a textúrou ovplyvňujú extrakciu čistej melódie. Pri historických nahrávkach je melódia, ako sme spomínali, veľmi často ornamentalizovaná a inak ľudovo alebo umelecky štylizovaná, čo znemožňuje extrakciu presných dát, ktoré by sme aj pri tých najlepších podmienkach z týchto nahrávok vedeli dostať.

Najlepšiu šancu programátorskej analýzy máme pri využití zápisov v zborníkoch. Napríklad notové zápisy vyššie spomínaného Karola Plicku zo zemplínskych oblastí počas svojich výskumov tento významný vedec vypracoval tak detailne, že ešte stále z nich folkloristi vyťahujú nepoznané piesne ako poklady z truhlice a spracuvávajú ich, či už v snahe napodobniť a zrekonštruovať pôvodný starobylý spev, alebo ho obohatiť a dať do nového šatu. Tieto piesne majú jediný problém – nie sú elektronizované.

Na Slovensku situácia elektronizácie archívnych materiálov, žiaľ, nie je v takom priaznivom stave, ako napríklad v Maďarsku, kde majú portál

`http://db.zti.hu/`,

na ktorom majú digitalizované tisícky historických fotiek, piesní a ďalších prameňov. Podobný projekt je Holandská zbierka *The Meertens Tune Collections*, ktorého tvorcovia na portáli

[1] Viac o tomto projekte `https://hc.sk/hudobny-zivot/clanok/historia/1314-nahravanie-folkloru-na-uzemi-slovenska-xv/?fbclid=IwAR1qHDjZDf0VOjEXhL6fXFmzmX9jGhe-a6RlbmMWYHRSrBx7BWsumqZA08Y`

http://www.liederenbank.nl/mtc/

centralizujú tisícky elektronizovaných vokálnych melódií z rôznych svetových databáz.

Našťastie máme na Slovensku dobrovoľníkov z projektu *Ludo Slovenský*[2]. Ľudo Slovenský je dobrovoľnícky projekt, ktorý sa snaží o centralizovanie databázy slovenských ľudových piesní. Piesne sú elektronizáciou rôznych historických prameňov, najmä zborníkov melódií zapísaných odborníkmi počas minulého storočia, napríklad zbierky [2]. V databáze sa už teraz nachádza viac ako 500 piesní a je predpoklad, že sa databáza v budúcnosti rozšíri. Okrem piesní sa v projekte zaoberajú aj ďalšími zaujímavými problémami, odporúčame čitateľovi navštíviť ich portál. Všetkým členom projektu patrí veľké poďakovanie za poskytnutie tejto databázy.

Ako obohatenie tejto databázy použijeme notové zápisy repertoáru FS Zemplín, ktoré spísal Ing. Radoslav Gajdoš, DiS.art.

[2]Viac o Ľudovi na https://ludoslovensky.sk/

# 4 Existujúce riešenia

## 4.1 Slovenský výskum

### 4.1.1 Výskumy týkajúce sa syntetickej analýzy slovenských ľudových piesní a výskumy s využitím výpočtovej sily

Pozrime sa na výsledky výskumov realizovaných na slovenských ľudových piesňach a špeciálne na výskumy využívajúce počítačové technológie.

#### Alica a Oskár Elschekovi

Na Slovensku sú v etnomuzikológii jednými z najvýznamnejších vedcov Alica a Oskár Elschekovi. Tí sa už dlhé roky venujú klasifikácii slovenskej ľudovej piesne a popísaniu jej atribútov. Vo svojej práci využívajú výsledky práce mnohých (celosvetovo) uznávaných vedcov, zberateľov a bádateľov, menovite napríklad H. Riemann, H. Schenker, H. Gräbner, L. Riemann, B. Bartók, K. Medvecký, K. Plicka, J. Kresánek a mnoho ďalších. Relevantným zdrojom bude pre nás zhŕňajúci článok [1], kde popisujú metodiku hudobných analýz folklórnych prejavov ako rozbor týchto stránok ľudovej piesne: tonalita, tónina, melodika, rytmika, metrika, forma, prednes, tempo. Popisujú aj vertikálnu analýzu pri viachlasnej piesni, tou sa však zaoberať nebudeme.

#### Výsledky za pomoci počítača *MSP 2A*

Základy automatizovaného výskumu folklórnych piesní sa položili už v polovici minulého storočia. Zhrnutie spolupráce českých a slovenských vedcov zo 70. rokov uplynulého storočia na tejto problematike popísal Ľubomír Chalupka v [7]. V jednej z popisovaných vedeckých prác išlo o experiment, aby sa zistilo, do akej miery dokáže počítač *MSP 2A* („‚malý samočinný počítač druhej generácie‘ vyrobený na Slovensku (konkrétne v Liptovskom Hrádku)“ (str. 5)) heuristickým programovaním vykonať tonálnu a formálnu analýzu 758 piesní zo Záhoria, ktoré zozbieral Janko Blaho. Práca zhŕňa, ako sa vedcom podarilo automatizovane zostaviť požadované intervalové štruk-

túry (kostry), vytvoriť frekvenčné poradia jednotlivých tónov, rozpoznávať, či je pieseň v durovej (iónskej), molovej (aiolskej) tónine, prípadne v inom modálnom type (dórsky, frygický, lydický a myxolydický) alebo dokonca v molovej melodickej a molovej harmonickej tónine (v piesňach v tónine **C** sa často vyskytoval klaster **G As A B H C**). Za spomenutie stojí aj automatizované identifikovanie formovej schémy ($AA$, $AB$...).

### Počítačová podpora EM

Pri automatizovanej analýze sa v článku [3] zaoberali nahrávkami piesní. Pre každú nahrávku vytvorili oscilogram, spektrálnu analýzu, spektrogram, sonogram, melogram a analýzu intenzity zvuku. Pre náš problém sú to, žiaľ, nevyužiteľné výskumné výsledky, pretože naša databáza je čisto symbolického charakteru.

Pozrime sa na štatistickú analýzou piesní v článku. Diagramovo popísali (str. 3 a 6) analytický prístup manželov Elschekovcov do chronologického postupu, ktorý je rozdelený do 22 fáz. Každá fáza sa zaoberá inou vlastnosťou piesne, postupne si ich popíšeme.

*Pozn.* Muzikologické pojmy vysoko presahujú muzikologickú prípravu na začiatku práce. Po popísaní celého postupu ho v jednoduchosti zhrnieme a zanalyzujeme všeobecný charakter tohto výskumu a jeho dopad na našu prácu.

1. Začiatok rozboru
2. Ambitus
   - Určíme tónový rozsah celej melódie.
3. Finálny tón
   - Posledný tón celej piesne.
4. Centrálny kostrový tón
   - Určíme centrálny tón kostry, je pevným jadrom melodiky, okolo ktorého sa odohráva pohyb. Je najdôležitejším tónom tonálnej kostry.
   - Znaky pre určenie centrálneho tónu:
     (a) chromaticky sa nemení (platí vždy),
     (b) býva totožný s finálnym tónom piesne,
     (c) býva koncovým tónom melodických riadkov, fráz,
     (d) má obvykle dlhšiu rytmickú hodnotu ako iné prechodné, mimokostrové tóny (platí hlavne v parlandových piesňach),
     (e) vyskytuje sa na silných (ťažkých) alebo zdôrazňovaných dobách (platí hlavne v rytmicky viazaných piesňach),

(f) melodika sa k nim najčastejšie vracia (platí často).

5. Tónový rad (tónina)
   - Určíme tóninu piesne.
6. Ďalšie tóny tonálnej kostry
   - V tónovom rade nájdeme ďalšie tóny tonálnej kostry. Sú to základné piliere melodickej stavby:
     (a) melodika sa k nim častejšie vracia,
     (b) bývajú zdôraznené fermátami,
     (c) tvoria ich často finálne tóny melodických riadkov.
7. Tonalita
   - Určíme tonálnu skupinu (typ tonality), do ktorej pieseň patrí, podľa kritérií:
     (a) intervalová vzdialenosť kostrových tónov,
     (b) počet kostrových tónov,
     (c) počet tonálnych kostier.
8. Melodická kostra
   - Určíme melodickú kostru, ktorú tvoria jednak tóny tonálnej kostry, jednak ďalšie tóny, podľa kritéria:
     (a) melodický pohyb sa k nim často vracia.
9. Grafický obraz melódie
   - Vykreslíme v notovej osnove grafický obraz melódie pre všetky melodické riadky (MR), do obrazu vyberieme tóny podľa kritérií:
     (a) počiatočné tóny MR,
     (b) vrcholy melodického priebehu MR,
     (c) finálne tóny MR.
   - Vynecháme nepodstatné prechodné tóny melódie.
10. Označenie finálnych tónov
    - Označíme finálne tóny melodických riadkov (MR)
      (a) melodický pohyb sa k nim často vracia.
11. Charakteristika melodiky

- Určíme prevažný smer melodiky (stúpajúca, klesajúca, ...).

12. Forma

    - Každý MR označíme veľkým písmenom abecedy (A, B, C...), pričom každé nové písmeno označuje melodickú frázu (MF) s novým hudobným obsahom po melodickej či rytmickej stránke.
        (a) Pri určení formy po melodickej (intonačnej) stránke prihliadame iba na melodickú líniu.
        (b) Pri určení formy po rytmickej stránke berieme do úvahy len rytmický pôdorys MR.

13. Výsledná forma

    - Určíme výslednú formu (ak je intonačná forma $ABC$ a rytmická $ABA$, výsledná forma bude $ABC$).

14. Charakteristika formy

    - Určíme podľa spôsobu vzniku formy, napríklad:
        (a) p – forma vzniknutá opakovaním ($|:A:|$),
        (b) o – forma otvorená (napr. $ABC$),
        (c) u – forma uzatvorená ($ABA$),
        (d) z – forma vzniknutá združovaním ($ABBC$).

15. Charakteristika metrickej stavby

    - Určíme podľa počtu slabík vo veršoch piesne. Napríklad izometrická stavba znamená:
        (a) (im) – všetky verše majú rovnaký počet slabík.

16. Charakteristika podickej stavby

    - Určíme podľa počtu taktov MR. Napríklad heteropodická stavba (hp) znamená rozdielny počet taktov v každej MF.

17. Charakteristika rytmu a rytmická klasifikácia

    - Charakteristiku rytmu určíme podľa rytmického obsahu jednotlivých MR. Napríklad birytmická stavba znamená:
        (a) (br) – v piesni sa vyskytujú dve odlišné rytmické jednotky.

- Rytmická klasifikácia sa určuje podľa charakteru rytmických hodnôt v takte, napríklad deliteľný rytmus:

  (a) (dr) – základné hodnoty sú symetricky deliteľné.

18. Tempo

    - Určíme metronómové tempo (MM). MM je relatívne tempo, určíme ho podľa predpokladaného počtu štvrťových hodnôt za jednu minútu[1].

19. Melodický incipit

    - Označíme tóny melodickej línie v prvom melodickom riadku (MR). Robíme to podľa Bartókovho systému (Nad CT arabskými číslicami, pod CT rímskymi číslicami. Centrálny tón označujeme číslicou 1.

20. Text

    - Určíme príležitosť (pri práci...), tematiku (napr. „smútok za slobodou“), textový incipit (text na začiatku piesne v pôvodnom znení), druh piesne, rýmová stavba, poetický rozbor, motívy, slabičnú stavbu.

21. Provenienčné údaje

    - Zápis základných údajov o lokalite, prameni, zapisovateľovi, signatúre, speváko vi, rok zberu, zápisu, druh záznamu a pod.[2]

      (a) (im) – všetky verše majú rovnaký počet slabík.

22. Koniec rozboru

Ako prvý postreh vieme pozorovať podobnosť s predošlou analýzou za pomoci počítača *MSP 2A*, kde sa tiež analyzovali finálne, centrálne kostrové tóny, tóniny, tonality a melodická kostra. Je to aj tým, že obidve analýzy sa opierali o štúdie manželov Elschekovcov.

Popisovaný postup je v článku [3] znázorňovaný v diagrame ako chronologická postupnosť krokov, ktorá sa má vykonať ako predpríprava na štatistickú analýzu väčšieho datasetu piesní. Je to skvelý návod na analýzu slovenských ľudových piesní za pomoci apriori vypracovaných metadát. Ak by sme disponovali takýmito dátami, naše analýzy by mohli byť presnejšie a veľká časť analýzy, ktorú po behu algoritmu musia vykonať muzikológovia, by sa dala eliminovať. Zároveň by sa dali využiť sofistikovanejšie metódy analýzy označkovaných dát, ktoré boli vyvinuté v obore analýzy dát

[1] Vyššie v texte spomínané *BPM*.

[2] Vyššie spomínané metadáta.

(*data science*) a strojového učenia (*machine learning*). Osobitne sa tejto problematike venovali v článku [9]. Špeciálnu pozornosť by sme vedeli venovať napríklad bodu 20, keďže v bakalárskej práci sme sa venovali hĺbkovému porovnávaniu textov piesní.

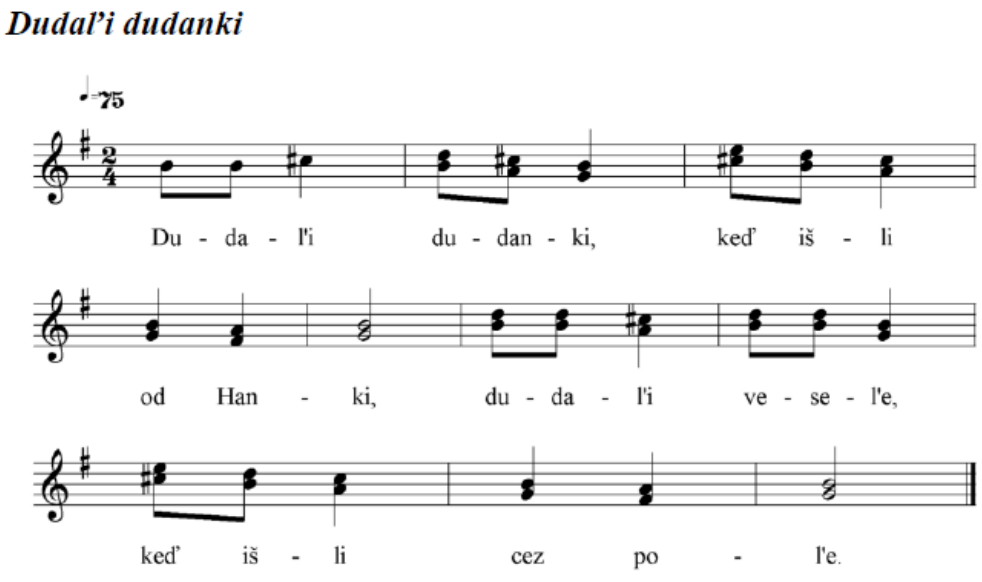
Zvažovali sme aj automatizované získavanie týchto dát. V prvotnom návrhu diplomovej práce bolo vytvorenie tabuľky pre každú pieseň a následné porovnávanie len týchto tabuliek. Dospeli sme však k záveru, že to robiť nebudeme. Prvý dôvod je, ako je vidieť v popise určovania týchto vlastností, že agregujú veľa menších kritérií veľmi špecifickým spôsobom a niektoré sa dokonca opierajú skôr o empirické znalosti muzikológov. To by stále nebola prekážka, pretože by sme vedeli automatizovane získať aspoň tie, ktoré sú exaktné. Problém by nastal pri komplikovanosti týchto pojmov, ktoré vyžadujú hlbokú znalosť hudobnej teórie, ktorou až do takejto miery nedisponujeme a odborníka na takú intenzívnu spoluprácu sa nám nepodarilo nájsť.

Za najväčšiu prekážku však považujeme to, že pri určovaní mnohých týchto vlastností je využívaná informácia o rozdelení piesne do melodických riadkov. Už pri predošlej analýze [7] tieto dáta k dispozícii mali – každá pieseň bola do systému zadávaná ako reťazec číslic, ktoré reprezentovali relatívne reprezentácie melódií (tie v práci využívame, popíšeme ich neskôr) a každý melodický riadok v rámci piesne bol ukončený číslom 99, celá pieseň číslom 999. My takéto dáta k dispozícii mať nebudeme, keďže dáta využívané v tejto štúdii nemáme k dispozícii a nemáme ani informáciu o tom, že takto digitalizované dáta sú niekde na verejnom úložisku.

Preto sme sa rozhodli pre algoritmus *nezávislý od metadát piesní*.

### Výskum na Kysuckých piesňach

Príklad využitia štrukturálneho spracovania hudobných materiálov manželov Elschekovcov sme našli aj v [5]. Príklad analýzy piesne:




| Názov piesne: Číslo piesne: 62. | | Lokalita: | | Žáner: | |
|---|---|---|---|---|---|
| *Dudaľi, dudanki* | | Riečnica | | k tancu "Stolkovi" | |
| **Tonalita, tónina:** | **Ambitus:** | **Incipit:** | | **Úvodný interval:** | **Záverečný interval:** |
| kvintakordálna, lydická | 00/09<br>v. 6 | 04,04,06\|07,06,04 | | v. 3 | v. 3 |
| **Tempo:** | | **Metrum:** | **Vrcholný takt:** | **Rytmus:** | jednoduchý |
| 4 = 75 | | 2/4 | 3,8 | 8,8,4\|8,8,4 | |
| **Forma:** | | **Počet taktov:** | **Počet veršov a slabík v jednej strofe:** | | |
| A, A´<br>5 5 | | 10 | 4, 24 (6+6+6+6)<br>12+12<br>5 + 5 | | |

Obr. 9: Príklad analýzy piesne v [5]

Môžeme sa pozrieť napríklad na zahrnutie parametra žánru ľudovej piesne. Ten istým spôsobom určuje viacero vlastností piesne. Jednou z nich je napríklad jej tempo. Určiť tempo presnou jednotkou *BPM* je pri ľudových piesňach nemožné vzhľadom na to, že ľudové piesne nie sú naviazané na konkrétne inštancie ich prevedenia. Piesne sa tradovali ústnym podaním a týmto sa nevytvoril priestor na exaktné ustálenie tempa piesne, ale iba rámcového charakteru skladby. Tempo piesne je teda určené v mnohých prípadoch len jej žánrom. V tomto konkrétnom prípade veľmi špecifickým žánrom piesne – tanec „Stolkovi“. Iné, všeobecnejšie žánre ľudových piesní, sú napríklad rýchle

čardáše, voľné parlandovité piesne alebo krucene, ktoré sú známe svojím pokojným charakterom. Je dôležité povedať, že nie všetky žánre sú prepojené s tempom.

Veľmi často sa stáva, že súčasné kapely transformujú všetky piesne na tanečné a zrýchľujú ich tempo, aby ich mohli hrávať na zábavách, plesoch a svadbách. Stáva sa aj to, že tanečná pieseň sa zrýchli natoľko, až na ňu nie je možné tancovať v štýle, v ktorom sa na ňu pôvodne tancovalo. V štylizovanom folklóre je častý aj opačný jav – baladizovanie rýchlych piesní. Príkladom tohto javu je spracovanie piesne *Červene žeľene* vo filme *Pásla kone na betóne*. Touto transformáciou sa ľuďom vzdiaľuje pôvodný žáner piesne. Napriek tomu by porovnávanie žánrov piesní bol určite faktor, ktorý by pomohol presnosti algoritmu.

Definovať žáner piesne nejde inak ako manuálne a rovnako aj iné atribúty v tomto výskume boli vypĺňané ručne. Tomu sa chceme vyhnúť a týmto je to pre nás nevyužiteľné.

## 4.2 Zahraničný výskum

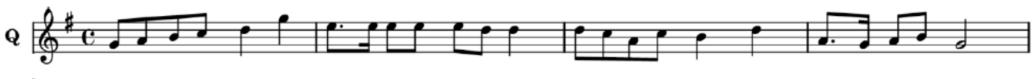
### 4.2.1 Spolupráca informatiky a muzikológie

Na existujúce riešenia sme nahliadli aj do zahraničia. Začneme článkom [8], v ktorom je dopodrobna popísaný problém melodickej variácie. Cieľom článku je popísať dôležitosť spolupráce medzi vedeckými oblasťami *MIR* (*Music information retrieval* – získavanie informácií z hudby), *CM* (*Computational musicology* – počítačová muzikológia) a *FSR* (*Folk song research* – výskum folklórnej piesne). Rozoberajú sa v ňom mnohé, v tom čase ešte otvorené, problémy. Napríklad absencia všeobecnej teórie ústneho podania piesní a jej prepojenie s inými hudobnými kognitívnymi procesmi ako ukladanie piesní do (ľudskej) pamäti, spievanie piesne spamäti a vytváranie novej piesne.

Ďalší popisovaný problém je konkrétnosť už vybudovaných modelov – všetky systémy, ktoré vyhľadávajú varianty sú programované s tým, že programátor dôverne pozná databázu a vie vďaka tomu algoritmus prispôsobiť čo najlepšie pre svoje potreby. Dôsledkom je to, že zatiaľ neexistuje všeobecné riešenie. Tento fakt sme si vzali do úvahy a náš softvér sme vyvíjali tak, aby bol čo najmenej závislý od dát a aby fungoval čo najprecíznejšie pre čo najširšie spektrum vokálnych melódií bez potreby ručne vypĺňaných metadát.

### 4.2.2 Reprezentácia dopytu ako vstupu pre vyhľadávací systém (*search engine*)

Článok [8] zhŕňa aj typy reprezentácie dopytov pre rôzne vyhľadávacie systémy. Keďže dopyt reprezentuje melódiu, znamená to prevedenie melódie do reťazca. V ilustračnom príklade v kapitole 3 z článku [8] sú na jednom mieste zhrnuté jednotlivé výsledky dopytu $Q$ nájdené rôznymi systémami v rôznych databázach. Pozrime sa postupne na všetky prístupy reprezentácií tejto konkrétnej melódie ako dopytu pre jednotlivé vyhľadávacie systémy, krátky popis systému a charakteristiku dát, ktorými disponovali tieto archívy v čase vydania článku.



Obr. 10: Ilustračný dopyt Q v článku [8]

#### Dánske folklórne archívy – *Danish Folklore Archives*

Dánske folklórne archívy – *Danish Folklore Archives* obsahovali okolo 10000 melódií zo zbierok a prepisov dánskej folklórnej hudby. Melódia je reprezentovaná ako reťazec prízvučných tónov, vzhľadom na charakter severských melódií je považovaná za prízvučnú prvá doba v takte. Tieto tóny sú reprezentované ako stupeň v rámci tóniny, čo nám dá sekvenciu

```
13516665
```

pre prvých osem dôb. Je to istá variácia reprezentácie relatívnych vzťahov tónov, ktorú sme spomínali v úvode článku. Rozdiel je len v tom, že sa veľkosť intervalu pripočíta alebo odčíta k predošlej hodnote a počiatočná hodnota je stupeň tónu v rámci tóniny (nie chromatickej stupnice).

#### Digitálny archív fínskych folklórnych piesní – *Digital Archive of Finnish Folk Songs*

Archív obsahuje okolo 9000 melódií, väčšinu z nich vyzbieral na začiatku dvadsiateho storočia Ilmari Krohn. Tu je melódia reprezentovaná ako sekvencia relatívnych vzťahov, čo nám dá postupnosť

```
+2+2+1+2+5-3
```

pre prvých 7 tónov. Druhá možnosť je reprezentovanie kontúrou melódie v Parsonsovom kóde – `u` (z angl. *up* – hore), keď melódia stúpa, `d` (z angl. *down* – dole), keď melódia klesá a `r` (z angl. *repeat* – opakovať), keď melódia ostáva na tom istom tóne. Stúpanie melódie znamená zmenu jej výšky o kladný interval (v relatívnej notácii $+i$), klesanie o zmenu výšky o záporný interval ($-i$) a opakovanie, ak melódia ostane na tom istom tóne ($+0$).

#### *Themefinder*

Systém *Themefinder* nie je priamo určený na analýzu folklórnych melódií, ale obsahuje aj veľké množstvo ľudových melódií. Databáza obsahuje veľkú zbierku tém a incipítov klasických skladieb, folklórnych piesní (kolekcia *Essen*, Schafftrath 1995) a Latinské motetá[3] zo šestnásteho storočia. Jedna možná reprezentácia je na základe reprezentácie melódie cez stupne tónov v rámci tóniny:

```
1 2 3 4 5 1 6
```

pre prvých 7 tónov. Nemusí to byť prvých 7, systém sa dá nakonfigurovať aj na vyhľadávanie rôzne dlhých fráz na konkrétnom mieste v piesni. Pri všetkých príkladoch sa však využívali začiatočné tóny kvôli odporúčanej najvyššej úspešnosti. Druhá je obdobná reprezentácia kontúry ako v predošlom príklade a tretia je vylepšená reprezentácia kontúry, kde sa rozlišujú kroky a skoky. Kroky melódie sú malé intervaly, skoky väčšie.

#### *MELDEX*

Dopyt sa dá vygenerovať klávesmi na virtuálnych klávesoch alebo aj pískaním a hmkaním melódie do mikrofónu. Interne využíva absolútnu reprezentáciu melódií, teda pre dopyt $Q$ (tóny ***G A C D G E***) to je

```
g a b c' d' g' e'.
```

Apostrof reprezentuje oktávu tónu. Používateľ si môže vybrať, či interne systém použije intervalovú sekvenciu alebo kontúru melódie. Systém *MELDEX* podporuje aj textové vyhľadávanie.

#### Databáza *Musipedia* a *YahMuugle*

V tejto databáze sa používa podobná absolútna reprezentácia melódie rozšírená aj o trvanie tónov:

```
c"4 d"2 g"2 e"4.
```

[3] Viachlasná vokálna skladba v rýchlejšom tempe

#### Zhrnutie

V slovenskej piesni musíme brať do úvahy začiatky a konce všetkých melodických riadkov, rovnako ako aj rytmickú podobnosť piesní. V týchto piesňach vyhľadávali len piesne na základe krátkych dopytov, čo v našom prípade nepostačuje na hĺbkovú analýzu piesne. Vyplýva to však aj z charakteru týchto prác, pretože ich cieľom je navrhnúť čo najlepší vyhľadávací a nie porovnávací systém, ako to máme za cieľ my.

#### Problém testovateľnosti systému

Veľmi dôležitý je aj problém testovateľnosti týchto systémov. Ideálne testovanie porovnávacieho algoritmu je automatizovaná verifikácia korektnosti jeho výsledkov. V týchto prípadoch majú programátori týchto systémov k dispozícii manuálne vyplnené dáta o variantoch a melodických rodinách, čo im umožňuje automatizovanú verifikáciu systémov a ich algoritmov. Pri podobnosti piesní, ktoré nie sú manuálne označené za varianty, takáto možnosť neexistuje a nikdy ani bez manuálnej intervencie existovať nebude, pretože na overenie algoritmu určenému na identifikáciu podobností potrebujeme algoritmus určený na identifikáciu podobností. Jediný spôsob, ako testovať systém, ktorý sa zaoberá podobnosťou piesní, je prizvať odborníkov, ktorí budú manuálne systematicky prezerať všetky výsledky, ktoré im systém vráti. To je nesmierne zdĺhavé (na konci po naprogramovaní nášho algoritmu sme videli, že už len pri našom datasete o veľkosti 500 melódií počet piesní podobných na viac ako 60% bol veľmi veľký), a teda toto testovanie nikdy nebude takej kvality ako automatizované.

### 4.2.3 Formalizácia problému podobnosti piesní a algoritmický postup porovnávania

Kognitívne hľadisko podobnosti melódií je široká a veľmi živá vedecká oblasť. V článku [6] porovnávajú autori výsledky algoritmov s výsledkami cieľovej skupiny študentov muzikológie na datasete krátkych útržkov melódií populárnych piesní. Predstavíme si najúspešnejšie algoritmy, ktoré použili.

#### 4.2.3.1 Miera podobnosti a formalizácia problému

Na to, aby sme vedeli definovať, čo je miera podobnosti (angl. *similarity measure*), potrebujeme transformovať čisté hudobné dáta na dáta spracovateľné programátorsky. V článku [6] využili matematický model dvojíc $t_n$ a $p_n$, kde prvá zložka je bod v čase a druhá zložka je výška tónu (angl. *pitch*). Tieto dva komponenty nazývajú rytmus a výška-melódia (angl. *pitch-melody*). Väčšina použitých mier podobnosti pracuje oso-

bitne s prvou alebo druhou zložkou. Autori zdôrazňujú, že postupnosti $t$ a $p$ je užitočné vnímať ako n-dimenzionálne vektory v priestore alebo v programátorskejšom ponímaní n-dimenzionálne reťazce.

**Definícia**

Miera podobnosti $\sigma(m_1, m_2)$ je symetrické zobrazenie priestoru abstraktných melódií $M$ mapujúce dve melódie na hodnotu medzi 0 a 1, kde 1 znamená identitu. Prirodzene by malo byť normalizované tak, že pre jednu melódiu $m$ je $\sigma(m, m) = 1$. Musí byť nezávislá od modulácie (ak je výška počiatočného tónu rôzna, ale melódia zachováva celú intervalovú sekvenciu, je považovaná za identitu) a posunutia melódie v čase (tá istá melódia znejúca v rôznych časoch je stále považovaná za tú istú melódiu).

Štandardný postup návrhu miery podobnosti je taký, že najskôr sa nad melódiami vykoná transformácia do rytmických a melodických vektorov a potom sa aplikujú iné, sofistikovanejšie metódy transformácie, napríklad gaussifikácia alebo fuzzifikácia. Ako posledný krok sa využije štandardná korelačná metóda, napríklad vektorová korelácia alebo Levenshteinova vzdialenosť.

#### 4.2.3.2 Transformácie melodickej zložky

**Kontúrizácia**

Ako prvý prístup je v článku popisovaná kontúrizácia. Prístup je analogický ako v článku [8]. Idea tejto transformácie je založená na fakte, že pri vnímaní melódie je kognitívne kladený dôraz na zlomové body v melódii, nie na presnú postupnosť výšok tónov. Využívaný bol Steinbeckov model (1982), ktorý neberie do úvahy repetíciu (vyššie popisované `r` – repeat), ale iba stúpanie a klesanie melódie. V druhom, vlastnom modeli, už braná do úvahy je.

**Fuzzifikácia**

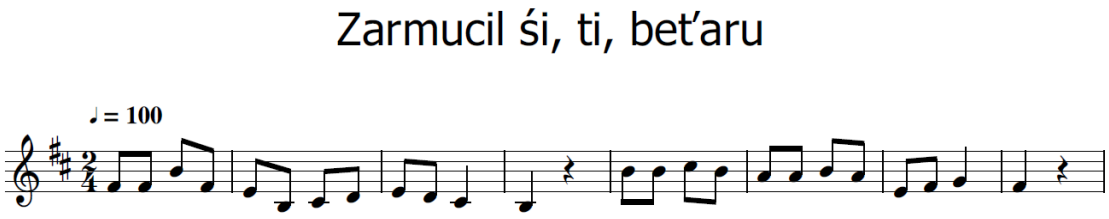
Druhý prístup je fuzzifikácia. Hlavná myšlienka fuzzy logiky je namiesto štandardných hodnôt 0 (*false*) a 1 (*true*) povoliť aj iné hodnoty medzi týmito hodnotami. Fuzzy množina (1965) je množina, do ktorej každý prvok patrí len s určitou mierou príslušnosti medzi 0 a 1. Výhoda tohto konceptu je, že umožňuje jednoduché modelovanie fuzzy vlastnosti v iných oblastiach. V oblasti podobností melódií sa dá tento prístup využiť pri klasifikovaní intervalov. Tento prístup sa opiera o fakt, že aj skúsený poslucháč nie vždy rozlíši presnú hodnotu intervalu (v intervalových sekvenciách, ktorými sme modelovali relatívnu reprezentáciu melódie, to znamená identifikovať presnú hodnotu medzi za sebou idúcimi tónmi melódie). Každý poslucháč však vždy odlíši „krok“

od „skoku“. Tento prístup využíval aj systém *Themefinder*, tu je však zovšeobecnený na fuzzy tabuľku intervalov. Každý interval je priradený do nejakej triedy ekvivalencie v tom zmysle, že všetky intervaly v rámci tejto triedy sú si v tejto transformačnej technike rovnocenné. Tabuľka vyzerá nasledovne.

| trieda | intervaly | názov |
|---|---|---|
| $-4$ | $< -7$ | veľký skok nadol |
| $-3$ | $-7, -6, -5$ | skok nadol |
| $-2$ | $-4, -3$ | malý skok nadol |
| $-1$ | $-2, -1$ | krok nadol |
| $0$ | $0$ | rovnaký |
| $1$ | $1, 2$ | krok nahor |
| $2$ | $3, 4$ | veľký krok nahor |
| $3$ | $5, 6, 7$ | skok nahor |
| $4$ | $> 7$ | veľký skok nahor |

Tabuľka 1: Fuzzy tabuľka melodických intervalov

Ilustrovať si tento systém môžeme na piesni z Veľkých Zalužíc *Zarmucil si, ti, beťaru* z repertoáru Folklórneho súboru Zemplín.




Obr. 11: Melódia piesne *Zarmucil śi, ti, beťaru*

Intervalová sekvencia prvých štyroch taktov tejto piesne je

`+0 +6 -6 -2 -6 +2 +1 +2 -2 -1 -2`,

po fuzzifikácii podľa tabuľky dostaneme intervalovú sekvenciu

`+0 +3 -3 -1 -3 +1 +1 +1 -1 -1 -1`.

#### 4.2.3.3 Transformácie rytmickej zložky

**Gaussifikácia**

Idea gaussifikácie spočíva v skonštruovaní spojitej integrovateľnej funkcie z rytmickej postupnosti. Náš n-dimenzionálny vektor rytmických hodnôt by však táto transformácia zmenila na nekonečný, čo by spôsobovalo v našej implementácii značné problémy. Rozhodli sme sa tento prístup z tohto dôvodu nepoužiť a teda ani bližšie nepopisovať.

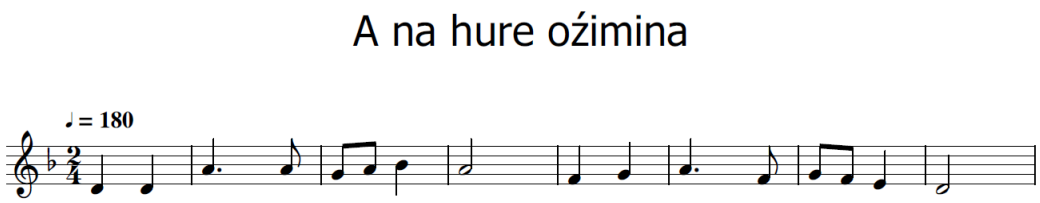
**Fuzzifikácia**

Fuzzifikácia rytmu je založená na rovnakej myšlienke ako fuzzifikácia melódie. V tomto článku[4] sa rozhodli dĺžky normalizovať modusom $m$ rytmu melódie, teda najčastejšou hodnotou v rytmickej postupnosti. Potom vytvorili nasledovnú tabuľku na kategorizovanie dĺžky podľa hodnoty $f_n = \frac{t_n}{m}$.

| trieda | interval hodnoty $f_n$ | trvanie |
|---|---|---|
| 4 | $(3, 3, +\infty)$ | veľmi dlhé |
| 3 | $(1, 8, 3, 3]$ | dlhé |
| 2 | $(0, 9, 1, 8]$ | normálne |
| 1 | $(0, 45, 0, 9]$ | krátke |
| 0 | $(-\infty, 0, 45]$ | veľmi krátke |

Tabuľka 2: Fuzzy tabuľka rytmických dĺžok

Ilustrovať tento prístup môžeme na útržku piesne *A na hure ožimina*.




Obr. 12: Melódia piesne *A na hure ožimina*

Rytmický vektor pre jednoduchosť vytvorme tak, že jedna doba bude mať trvanie 2. Takto dostaneme reťazec pre prvé 4 takty

2 2 3 1 1 1 2 4.

[4] [6]

Na fuzzifikáciu potrebujeme vyrátať *modus* tejto sekvencie – ako konkrétne túto hodnotu vyberať pri rovnakom počte dvoch hodnôt v článku popísané nie je, pre ilustratívnosť vyberieme hodnotu 2. Po predelení každej hodnoty týmto *modus*-om dostaneme sekvenciu hodnôt

1 1 1,5 0,5 0,5 0,5 1 2.

Po fuzzifikácii podľa tabuľky dostaneme sekvenciu indexov tried

2 2 2 1 1 1 2 3.

Tento prístup zjednotil štvrťovú notu so štvrťovou notou s bodkou[5], čo je v ľudovej piesni veľmi častý spôsob variácie. Treba povedať, že fuzzifikácia pri výbere hodnoty 1 by nemala žiaden efekt. Z hľadiska ľudovej piesne má určite najväčší zmysel vyberať namiesto modusu melódie hodnotu, ktorá reprezentuje jednu dobu v takte, čo v tomto prípade bola hodnota 2 a práve ňou sme dosiahli cielený efekt.

#### 4.2.3.4 Korelačné metódy

Ako korelačné metódy boli využité vektorové korelácie, *Sum Common* metóda, *Count Distinct* metóda, Ukkonenova miera a iné. Najefektívnejšia sa ukázala byť Levenshteinova vzdialenosť. Táto miera bola využitá presne tak ako v bakalárskej práci, ktorú táto práca rozširuje. V najjednoduchšej forme hovorí o tom, koľko zmien potrebujeme na to, aby sme zmenili jeden reťazec na druhý. Využili ju na reťazce čistých a kontúrizovaných melódií, intervalov (relatívna intervalová reprezentácia) a fuzzifikovaný rytmus. Načrtli aj veľmi zaujímavý model váženej vzdialenosti na základe harmonického vektora, implementácia by však presahovala rozsah tejto práce a ostáva len ako cieľ do budúcna.

[5] Bodka napravo od noty predlžuje trvanie tónu o polovicu jeho trvania – v prípade štvrťovej noty, ktorá trvá jednu dobu, predĺži jej trvanie na jeden a pol doby

# 5 Technológie

## 5.1 Programovací jazyk

Keďže bakalárska práca *Hľadanie podobností v textoch ľudových piesní* bola programovaná v jazyku Java, pokračujeme vo vývoji v ňom.

V bakalárskej práci sme mali zaužívané zvýrazňovanie textu, ktorý súvisí s programom takýmto `modrým fontom`. Budeme to robiť takto aj teraz. Tiež zachováme zaužívané označenia objektov, ktoré v nej boli navrhnuté.

## 5.2 Verzionovací systém

Naďalej sme rozširovali *Git* repozitár aplikácie *SimFolk* na doméne *GitHub*. Nájdete ho tu:

`https://github.com/MichalMizak/SimFolk`.

## 5.3 Technologické aspekty databázy melódií a formát MusicXML

Všetky dáta, s ktorými budeme pracovať, sú uložené v štandardizovanom formáte MusicXML[1]. Tento formát bol navrhnutý na archivovanie a zdieľanie symbolického notového zápisu medzi aplikáciami a nahrádza *MIDI* formát, oproti ktorému je podstatne obšírnejší a mocnejší (napriek tomu sa ešte stále používa, napríklad v spomínanej zbierke *The Meertens Tune Collections* a vygenerovaných piesňach spomínaných v sekcii o autorských právach). Prirodzene prevyšuje aj potreby našej práce (poskytuje napríklad aj viacstopový polyfónický zápis), je to však momentálne najsofistikovanejší formát na uchovávanie symbolicky znotovanej hudby. Existuje k nemu aj množstvo knižníc, jedna z nich je napríklad voľne dostupná Java knižnica *proxymusic*[2], pomocou

[1] `https://www.musicxml.com/`

[2] `https://github.com/Audiveris/proxymusic`

ktorej budeme serializovať a deserializovať objekty reprezentujúce melódiu do a z `.xml` súborov.

### 5.3.1 Deserializácia `.xml` súborov

Na deserializáciu `.xml` súborov použijeme knižnicu *proxymusic*. Knižnica pozostáva z *Java* tried, ktoré boli vygenerované podľa schémy `musicxml.xsd`[3], ktorá definuje *Musicxml*. Následne boli mierne upravené kvôli pohodlnejšiemu používaniu. Tieto triedy reprezentujú jednotlivé elementy v tejto schéme a zároveň zachovávajú hierarchickú štruktúru tejto schémy.

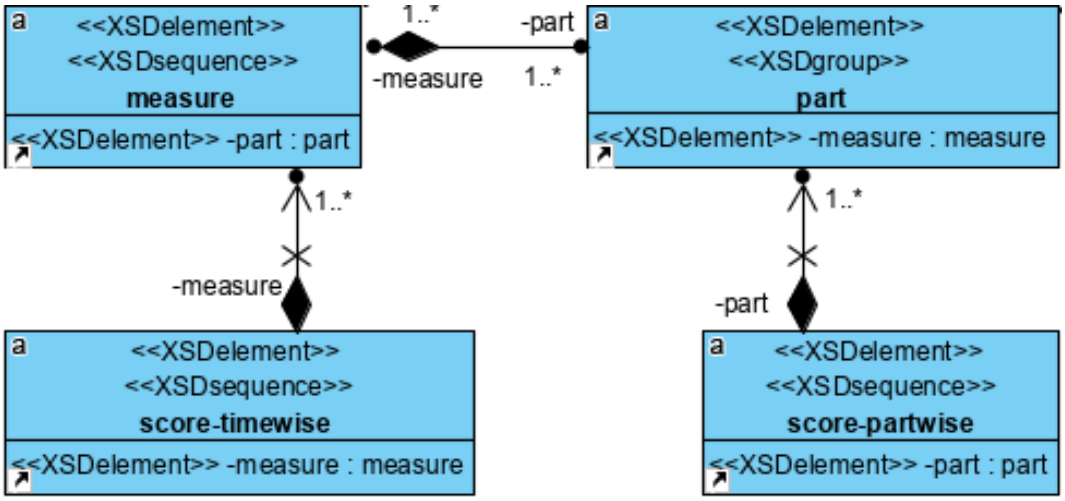
#### 5.3.1.1 Knižnica *proxymusic*

Do projektu sme knižnicu *proxymusic* importovali pomocou pridania závislostí v štandardnom *Maven* `pom.xml`. Ako základná súčasť knižnice figuruje trieda `Marshalling`, ktorá pomocou metódy `unmarshall` vráti „naplnený“ *Java* objekt dátami z jedného `.xml` súboru. K ďalším objektom sa programátorsky dá pristupovať celkom jednoducho a intuitívne cez *getter* metódy na jednotlivých inštanciách a týmto iterovať rekurzívnou štruktúrou schémy *musicxml*.

### 5.3.2 Štruktúra *Musicxml*

V našej práci budeme pracovať s typom *Musicxml* `ScorePartwise`, ktorý obsahuje zoznam partov[4] danej partitúry[5], ktorú daný `.xml` súbor reprezentuje. Sú to objekty typu `Part`, z ktorých každý jeden reprezentuje melodickú linku (aj polyfonickú = viachlasnú) zapísanú po taktoch. Existuje aj prístup `ScoreTimewise`, kde, naopak, čítame zoznam taktov všetkých partov. Chronologicky je teda pre skladbu zapísaný prvý takt pre každý part, po ňom druhý takt pre každý part, atď. V prístupe `ScorePartwise` nasledujú všetky takty pre prvý part, potom všetky takty pre druhý part atď. Graficky si môžeme tento rozdiel prezrieť na obrázkoch vygenerovaných pomocou nástroja *Visual paradigm*. Na obrázkoch sú útržky z obrovského *Class diagram*-u `musicxml.xsd`.

[3] `https://www.musicxml.com/for-developers/musicxml-xsd/`
[4] Notový zápis pre 1 nástroj v skladbe
[5] Notový zápis všetkých hlasov hudobnej skladby




Obr. 13: `Partwise` vs. `Timewise`

Vidíme, že k elementu `part` je nalinkovaný zoznam elementov `measure`, ktoré reprezentujú jednotlivé takty partu. Naša databáza je homogénna v tom, že obsahuje len jediný „nástroj“ a to práve vokálny, čo značne zjednodušuje prácu s týmito dátami a aj preto volíme prístup `ScorePartwise`. Takt je nalinkovaný na objekt `music-data`, ktorý je generickým objektom na popisovanie vlastností hudobných objektov tejto schémy. V prípade taktu obsahuje vlastnosti (anglicky *properties*) a všetky objekty nachádzajúce sa v takte. Z týchto objektov je pre nás najzaujímavejší zoznam objektov `note`, ktoré reprezentujú noty a hudobné pomlčky (reprezentujúce „prestávku“). K tomuto objektu je nalinkovaný objekt *pitch*, ktorý obsahuje atribút *step*, ktorý hovorí o výške tónu a atribút *duration*, ktorý hovorí o dĺžke – rytmickej informácii tónu.

Pre potreby našej práce v našom kóde nebudeme musieť naprogramovať *DTO*[6] na vytvorenie vlastných objektov, ale postačí nám ich konverzia na `String`, ktorá tvorí základný pilier našej práce. Bližšie ju popíšeme v ďalších kapitolách.

[6] Data transfer object – objekt využívaný na zapuzdrenie dát

# 6 Návrh a implementácia riešenia

V tejto kapitole prejdeme celým postupom algoritmu, ktorý má základ v algoritme bakalárskej práce, ktorú táto práca rozširuje. Na tento algoritmus sa budeme odvolávať ako na pôvodný. Pre úplné pochopenie je nutné poznať pôvodný algoritmus, budeme sa však snažiť o jeho zjednodušené vysvetlenie a uchopenie praktického efektu tej-ktorej časti algoritmu.

## 6.1 Načítanie dát

Pri načítaní dát najprv zvolíme vstupný zoznam `.musicxml` (alebo `.xml`, ak zachovávajú schému `musicxml.xsd`) súborov, ktoré chceme spracovávať – piesne, ktoré chceme porovnávať. Nasleduje načítanie dát do Java objektov. Po načítaní máme všetky potrebné údaje o piesni v objekte typu `MelodySong`, ktorý rozširuje pôvodný typ `Song` a obsahuje navyše zoznam taktov piesne a súbor typu `File`, z ktorého bol vytvorený. Ako sme spomínali, databáza obsahuje výlučne jednohlasné piesne v jednom parte. S týmito taktami pracujeme ako so zoznamom objektov typu `Measure`, presnejšie `ScorePartwise.Part.Measure`. Dáta z *Musicxml* formátu sme spracovávali pomocou knižnice *proxymusic*, z ktorej táto trieda pochádza.

## 6.2 Problém premazania databázy

V bakalárskej práci boli všetky dáta ukladané do databázy *MySQL*[1]. Cieľom tohto prístupu bolo ukladať čo najviac medzivýsledkov kvôli spätnej kontrole a analýze algoritmu. Tento prístup sme zachovali.

Predošlý algoritmus umožňoval spustenie porovnávania piesní, v tom čase len ich textu, so všetkými piesňami v databáze bez toho, aby boli uložené do databázy. Táto funkcionalita mala svoje výhody, ale pri zovšeobecňovaní algoritmu sme došli k záveru, že ju budeme musieť kvôli jednoduchosti odstrániť. Videli sme to jasne, keď sme

[1] Vytvorenej podľa *create* skriptu `https://github.com/MichalMizak/SimFolk/blob/master/SimFolk/other/create_script.sql`

si dali do pomeru užitočnosť tejto funkcionality s technickými komplikáciami, ktoré spôsobovala. Pre 100% korektnosť výsledkov totiž bolo aj v minulom prístupe nutné púšťať algoritmus na prázdnej databáze, bez ukladania išlo len o akúsi aproximáciu výsledku. Urobili sme teda rozhodnutie, ktoré nás pred každým behom núti vymazať všetky dáta – funkcionalitu porovnávania v pamäti bez databázy sme odstránili. Samozrejme, jedná sa len o medzivýsledky, ktoré sa popri algoritme vyrátavajú – piesne v databáze sa mazať nemusia. Tento prístup zabezpečí konzistentné výsledky algoritmu a používateľovi nespôsobí veľké problémy. Okamžitý (lepšie povedané rýchly) výsledok sme zabezpečili iným prístupom – agregáciou medzivýsledkov. Spracovanie a ukladanie výsledkov sme urobili rafinovanejšie ako v pôvodnom algoritme, dostaneme sa k nemu v ďalšej kapitole.

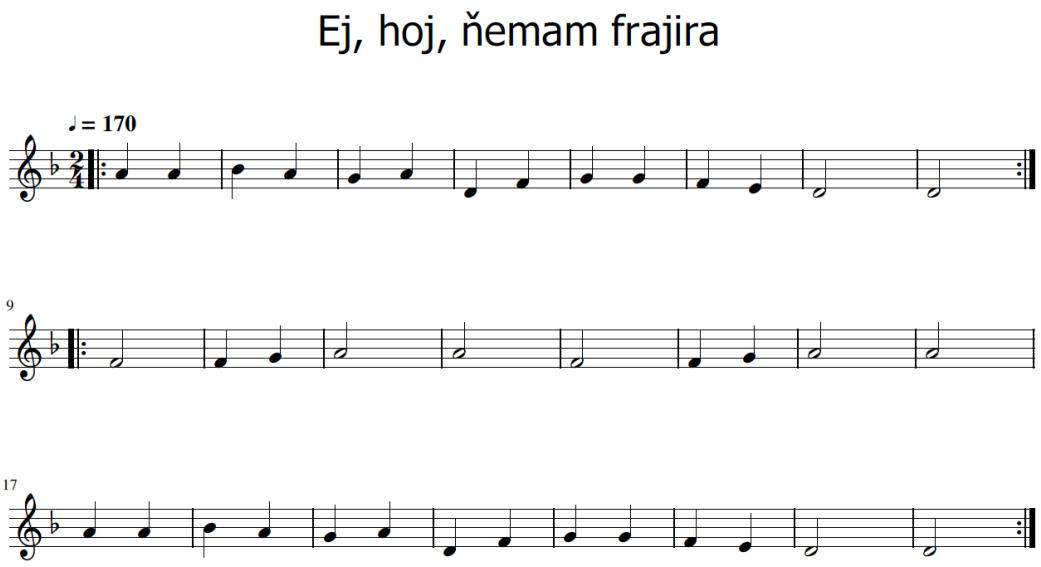
## 6.3 Využitie databázy

Po načítaní dát nastáva prvá fáza algoritmu – inicializácia metadát. Metadátami nazývame všetky dáta, ktoré sú potrebné na beh algoritmu, nie sú však výsledkom algoritmu. De facto sú v našom algoritme všetky dáta, ktoré boli vytvorené v pôvodnom algoritme, metadátami[2]. Týmto celú relačnú *MySQL* databázu považujeme len za databázu medzivýsledkov. Zhrnieme si, čo tieto metadáta znamenajú v algoritme spracúvajúcim melódie piesní.

## 6.4 Serializácia melódie do reťazca

Keďže musíme pri popise algoritmu postupovať chronologicky, začneme najväčšou a najdôležitejšou modifikáciou pôvodného algoritmu – serializáciou melódie do reťazca. Keďže algoritmus bol určený na porovnávanie textov, museli sme ho prepracovať tak, aby dokázal pracovať s melódiami. Problém, nad ktorým sme sa museli zamyslieť, bola komunikácia s databázou. Unikátnosť záznamov sa určovala na základe rovnosti reťazcov a to znamenalo, že ak by sme chceli zabezpečiť unikátnosť melódií, museli by sme prerobiť celý systém na porovnávanie komplikovanejších záznamov. Našli sme iné, jednoduchšie riešenie – prevod melódie na reťazec. V analýze zahraničných riešení sme vypísali pár spôsobov, ako to robiť. Vybrané spôsoby ilustrujeme na príklade piesne *Ej, hoj, ňemam frajira*.

[2] Netreba si tento pomýliť s metadátami piesne.




Obr. 14: Zemplínska pieseň z obce Pozdišovce *Ej, hoj, ňemam frajira*

### 6.4.1 Absolútna reprezentácia melódie

Prístup absolútnej reprezentácie melódie, ktorý sme si zvolili, spočíva v označení všetkých 12 tónov znakmi nasledovne:

| Tón | *C* | *Cis* | *D* | *Dis* | *E* | *F* | *Fis* | *G* | *Gis* | *A* | *Ais* | *H* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Znak | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B |

Tabuľka 3: Mapovanie tónov na znaky

Dôvod, že využívame znaky `A` a `B` a nie čísla `10` a `11` je, aby bol každý tón rovnocenný z hľadiska dĺžky vytvoreného reťazca. Zároveň sme sa rozhodli ignorovať oktávu tónu. Je to preto, lebo zmena oktávy tónu v rámci identifikácie piesne je zanedbateľná. Zmena oktávy je dokonca nástroj, ktorý sa často využíva na vytvorenie variantov. Takto teda dva reťazce (respektíve ich časti) zapísané v databáze v dvoch rôznych oktávach budú vyzerať rovnako. Pieseň *Ej hoj, ňemam frajira* vieme prepísať ako reťazec tónov (v slovenskej notácii a s pridanými medzerami po riadkoch kvôli čitateľnosti, v reálnej implementácii je to reťazec bez medzier) takto

`AABAGADFGGFEDD`
`FFGAAFFGAA`
`AABAGADFGGFEDD`

a potom po aplikovaní mapovania dostaneme

99A97925775422
5579955799
99A97925775422.

Je dôležité poznamenať, že vo všetkých konverziách zanedbávame opakovacie znamienka. Zmysel zamyslieť sa aj nad opakovaniami by malo vtedy, ak by sme mali k dispozícii formu piesne a rozdelenie do melodických riadkov.

### 6.4.2 Relatívna reprezentácia melódie

Relatívna alebo intervalová reprezentácia je práve relatívna intervalová sekvencia melódie. V tejto melódii by vyzerala takto

+0 +1 −1 −2 +2 −7 +3 +2 +0 −2 −1 −2 +0
+3 +0 +2 +2 +0 −4 +0 +2 +2 +0
+0 +1 −1 −2 +2 −7 +3 +2 +0 −2 −1 −2 +0.

### 6.4.3 Fuzzifikovaná relatívna reprezentácia melódie

Fuzzifikáciu sme adaptovali z článku [6]. Dostaneme reťazec

+0 +1 −1 −1 +1 −4 +2 +1 +0 −1 −1 −1 +0
+2 +0 +1 +1 +0 −2 +0 +1 +1 +0
+0 +1 −1 −1 +1 −4 +2 +1 +0 −1 −1 −1 +0.

Implementačne sme z praktických dôvodov triedy označili číslami 0–9.

### 6.4.4 Reprezentácia kontúry melódie

Reprezentáciu kontúry sme tiež adaptovali z článku [6], konkrétne ich vlastnú implementáciu, kde berú do úvahy aj repetíciu tónov. Dostaneme reťazec

rudduduurdddr
uruurdruur
rudduduurdddr

### 6.4.5 Fuzzifikovaná reprezentácia rytmu

Pri prezeraní nášho datasetu sme objavili menšiu komplikáciu v notácii rytmu plynúcu zo štandardu *Musicxml*. V tomto štandarde je adaptovaný spôsob zapisovania rytmu v *MIDI* formáte. To znamená, že na začiatku je v súbore zaznačená hodnota *divisions*, ktorá reprezentuje počet divízií prislúchajúcich jednej štvrťovej note. Trvanie

ostatných tónov je určované na základe tejto hodnoty. Napríklad pri hodnote *divisions* rovnej 4 by bola hodnota osminovej noty 2 a osminovej noty s bodkou 3. Toto však evidentne prináša nekonzistentné dĺžky tónov pre rôzne hodnoty *divisions*. Museli sme teda všetky piesne normalizovať podľa nejakej pevnej hodnoty. Zvolili sme si 24, pretože nám umožní zápis najmenej šestnástinových nôt a navyše aj zápis štvrťových a osminových triol[3]. Predpokladáme, že zložitejšie rytmické štruktúry sa v našej databáze v piesňach nenachádzajú.

Pri reprezentácii rytmického reťazca sme sa inšpirovali fuzzifikáciou v článku [6] a prispôsobili ju našim potrebám. Rozdiel je v tom, že na vyrátanie hodnoty $f$ nepoužívame *modus* všetkých rytmických hodnôt piesne, ale normalizujeme pomocou trvania jednej doby, čo je presne hodnota *divisions* rovná 24. V príklade *Ej, hoj, ňemam frajira* neuvídime význam tohto rozhodnutia, keďže jeho opodstatnenie je najmä v tom, že chceme zjednotiť noty s bodkovaným rytmom s nebodkovaným, a tak si uvedieme hodnotu $f$ pre najbežnejšie trvania tónov. Tieto trvania zapisujeme pomocou hodnoty *divisions* rovnej 24, hodnota $f$ vznikne predelením tohto trvania hodnotou 24 reprezentujúcou jednu dobu. Tóny sú zoradené podľa ich trvania a fuzzifikačnej triedy.

| typ noty | trvanie tónu | hodnota f | fuzzifikačná trieda |
|---|---|---|---|
| celá | 96 | 4 | 4 |
| polová s bodkou | 72 | 3 | 3 |
| polová | 48 | 2 | 3 |
| štvrťová s bodkou | 36 | 1,5 | 2 |
| štvrťová | 24 | 1 | 2 |
| osminová s bodkou | 18 | 0,75 | 1 |
| osminová | 12 | 0,5 | 1 |
| štvrťová triola | 8 | $0,\overline{33}$ | 0 |
| šestnástinová | 6 | 0,25 | 0 |
| osminová triola | 4 | $0,1\overline{66}$ | 0 |

Tabuľka 4: Príklad fuzzifikovaných tónov

Z ukážkovej piesne dostaneme po fuzzifikácii reťazec

22222222222233

3223332233

22222222222233

[3] Triola predstavuje odchýlku od pôvodnej rytmickej štruktúry skladby spočívajúcu v tom, že v rámci jednej rytmickej skupiny (napríklad jednej doby, ale nie je to podmienka) budú namiesto predpísaného počtu dielčích jednotiek odohrané tri jednotky

Spojením fuzzifikácie s rytmickou a kontúrovou reprezentáciou dokážeme spoľahlivo detekovať napríklad problém kolísavej kvarty.

## 6.5 Rozdelenie melódií do menších fráz

Pôvodný algoritmus každý vstupný text rozdeľoval do menších častí. Najpoužívanejšou metódou bolo rozdeľovanie do N-gramov – susedných postupností $n$ slov textu[4]. V algoritme pre melódie sme využili analogický prístup rozdeľovania melódií, respektíve rozdeľovania reťazcov, do ktorých bola melódia transformovaná. Konverzia do reťazcov a rozdeľovania melódií sa v praxi prelínajú a dejú takmer naraz, dajú sa však konfigurovať osobitne.

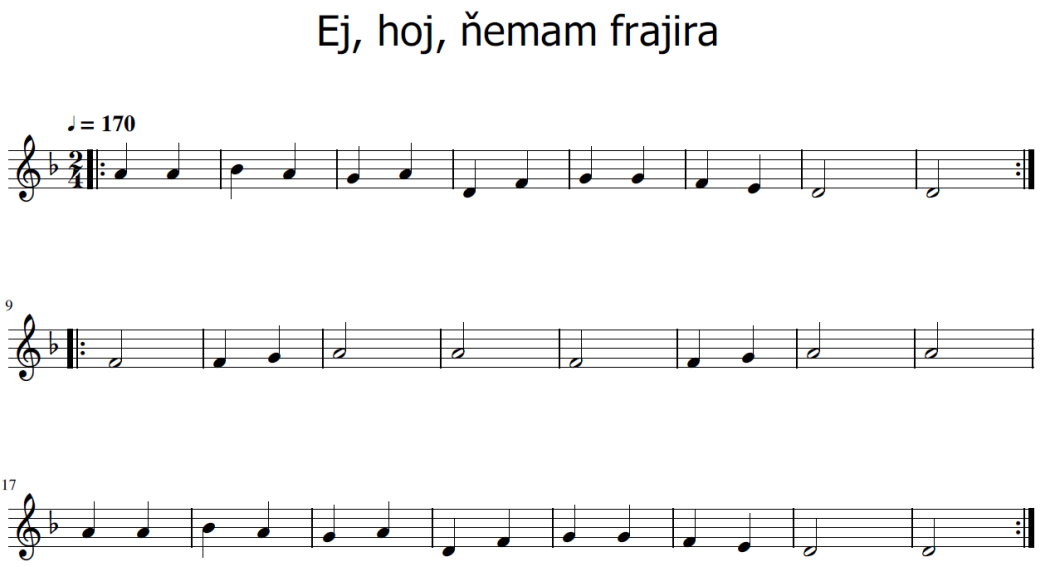
Pri vytváraní analógie k N-gramom je viac možností, ako melódiu rozdeľovať. Tieto možnosti závisia od najmenšej jednotky, podľa ktorej budeme N-gramy vytvárať. Pri porovnávaní textov si túto jednotku vieme predstaviť napríklad ako slovo, pri porovnávaní slov ako jedno písmeno. Intuitívne existujú dve najvhodnejšie možnosti – jeden tón alebo jeden takt. Vziať jeden tón za základnú jednotku N-gramov by malo zmysel pri porovnávaní parlandovitých piesní, kde neexistuje pevná štruktúra notácie a zápis melódie prudko závisí od jej prevedenia spevákom. Charakter databázy je však len zo zanedbateľne malej časti parlandovitý, väčšina piesní je pevne naviazaná na rytmickú štruktúru, ktorá sa zapisuje pevne do taktov. Týmto sme s ohľadom na dáta, efektivitu a úspešnosť algoritmu túto metódu nevyužívali, metóda je napriek tomu implementovaná pre prípad rozšírenia databázy.

Zo všetkých dĺžok n-gramov sme si za $n$ zvolili len 1 a 4. Vybrali sme práve tieto štruktúry, lebo 4-gram je najčastejšia dĺžka melodických riadkov. Frázy nepárnej dĺžky nie sú až tak bežným javom. Použitie N-gramov vo všeobecnosti zabezpečí zachovanie kontextu melodických fráz. Un-gram ($n = 1$) zasa umožní nájsť menej zrejmé podobnosti melódií a identifikuje jav vložených taktov do melódie. Týmto pomáha identifikovať varianty, ktoré vznikli týmto procesom variácie. Pri zemplínskych piesňach je časté napríklad vkladanie citoslovca *hojaja* na koniec frázy melódie, ktorá tento pokrik pôvodne nemala. Tento dodatok má často dĺžku jedného alebo dvoch taktov.

Pozrime sa na výsledok po rozdelení príkladnej piesne *Ej, hoj, ňemam frajira* do 4-gramov. Melódia je transformovaná na reťazec pomocou absolútnej reprezentácie.

*Pozn.* druhý riadok v reťazci reprezentuje 4-gramy, ktoré začínajú v prvom riadku a končia v druhom. Analogicky to platí pre štvrtý riadok. Takty sme oddelili vertikálnou čiarou kvôli čitateľnosti.

[4] Ak ho nepozná, odporúčame čitateľovi v krátkosti naštudovať tento známy koncept kvôli lepšiemu pochopeniu nasledujúceho odstavca.




Obr. 15: Zemplínska pieseň z obce Pozdišovce *Ej, hoj, ňemam frajira*

```
AA|BA|GA|DF BA|GA|DF|GG GA|DF|GG|FE DF|GG|FE|D GG|FE|D|D
FE|D|D|F D|D|F|FG D|F|FG|A
F|FG|A|A FG|A|A|F A|A|F|FG A|F|FG|A F|FG|A|A
FG|A|A|AA A|A|AA|BA A|AA|BA|GA
AA|BA|GA|DF BA|GA|DF|GG GA|DF|GG|FE DF|GG|FE|D GG|FE|D|D
```

Melódiu v práci rozdeľujeme do štruktúr jedného taktu (un-gramu), 4-gramu a celej melódie (tento prístup využívame kvôli overeným výsledkom algoritmu *edit distance*).

## 6.6 Zgrupovanie do tried ekvivalencií

Najdôležitejšia časť predošlého algoritmu bolo zgrupovanie reťazcov do tried ekvivalencií. Vstupnými parametrami tejto fázy je informácia o miere tolerancie a porovnávacom algoritme, podľa ktorého budeme reťazce stotožňovať. Môže ísť o ľubovoľný algoritmus. My sme si zvolili algoritmus *edit distance*, ktorý je najčastejším algoritmom na porovnávanie dvoch reťazcov. Tolerancia môže byť žiadna, nízka, stredná alebo vysoká. Reprezentujeme ju ako konštantu, ktorej priradzujeme numerickú hodnotu z intervalu $[0, 1]$. Zvolili sme tabuľku

| tolerancia | numerická hodnota |
|---|---|
| HIGH | 0, 7 |
| MEDIUM | 0,8 |
| LOW | 0,9 |
| NONE | 1 |

Tabuľka 5: Tabuľka tolerancie

Dva reťazce sú najskôr porovnané normalizovaným algoritmom editačnej vzdialenosti a potom sa porovná výsledná hodnota s numerickou hodnotou tolerancie. Ak je vyššia, považujeme reťazce za podobné.

Reťazec pridávame do triedy ekvivalencie na základe dvoch kritérií: `MATCH ALL` alebo `MATCH ONE`. Prvé znamená, že reťazec musí byť podobný so všetkými reťazcami v danej triede, druhé znamená, že stačí, aby bol podobný aspoň s jedným reťazcom. Tento parameter je, samozrejme, konfigurovateľný. V tejto práci sme využívali prvé kritérium, pretože spôsobuje menej výrazné zgrupovanie krátkych fráz a vráti bezpečnejšie výsledky v tom zmysle, že vracia menej falošne pozitívnych podobností melódií.

Implementačný problém, ktorý sme riešili, je, že v predošlej fáze musíme zabezpečiť unikátnosť reťazcov – v celej databáze sa jeden reťazec vytvorený jednou stratégiou môže nachádzať iba raz. Musí teda nastať synchronizácia s databázou a generovanie unikátnych `id`. Potom nastáva ich systematické zgrupovanie do tried a každej piesni sa priradí celý zoznam takých tried ekvivalencií, ktoré obsahujú aspoň jednu frázu danej piesne. Triedam je tiež priradené unikátne id a v ďalších fázach sa porovnávajú len na základe neho. Tento proces sa deje na začiatku algoritmu pre všetky melódie a všetky vytvorené reťazce kvôli zabezpečeniu homogénneho zgrupenia všetkých reťazcov. Ďalšie fázy sa už dejú samostatne pre dvojice piesní s týmito vytvorenými metadátami, ktoré dopytujú z databázy.

### 6.6.1 Prvotná absolútna reprezentácia melódie

Pri popisovaní absolútnej reprezentácii melódie, ktorú sme využili, sme hovorili o tom, že potrebujeme, aby všetky tóny boli rovnocenné vzhľadom na dĺžku reťazca. Je to tak práve kvôli tomu, že v prvotnej verzii algoritmu sme všetky jednociferné čísla (poradia tónov v chromatickej stupnici) normalizovali na dĺžku dva pridaním nuly takouto tabuľkou:

| Tón | *C* | *Cis* | *D* | *Dis* | *E* | *F* | *Fis* | *G* | *Gis* | *A* | *Ais* | *H* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Znak | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 |

Tabuľka 6: Pôvodné mapovanie tónov na reťazce

Melódiu `A H C` by sme reprezentovali ako `091101`, melódiu `A Ais C` ako `091001` a melódiu `A G C` ako reťazec `090801`. Tieto melódie sa líšia v jednom tóne, čo nekorešponduje počtu znakov, ktorými sa líšia a to spôsobuje nepresnosť algoritmu editačnej vzdialenosti. Toto bol aj jeden z viacerých dôvodov, prečo sme vyradili zápis oktávy z absolútnej reprezentácie.

### 6.6.2 Porovnávanie intervalových reťazcov

Je vidieť, že zachovať dĺžku relatívnych reprezentácií melódie tak, aby počet rozdielnych znakov korešpondoval s reálnym počtom rozdielov, nie je možné. Museli sme teda pri porovnávaní takýchto reťazcov pristúpiť k riešeniu, že tieto reťazce rozdelíme a pretransformujeme na pole čísel. Napríklad zo začiatku príkladného reťazca

```
+0 +1 -1 -2 +2 -7 +3
```

sa stane nasledovný zoznam objektov typu `Integer`

```
[0, 1, -1, -2, 2, -7, 3].
```

Je evidentné, že tento prístup je pomalší, pretože musí dochádzať ku konverzii objektov typu `String` na objekty typu `Integer`. Nepovažujeme to však za problém vzhľadom na to, že kvalita nášho algoritmu nie je meraná jeho rýchlosťou, ale precíznosťou. Konečný algoritmus je totiž deterministický – stačí ho spustiť raz a výsledky ostávajú nemenné.

### 6.6.3 Vylepšené edit distance

Pri implementácii sme sa zamýšľali aj o využití vylepšenej implementácie s tabuľkou definujúcou podobnosť jednotlivých tónov, a teda *pravdepodobnosť zámeny* týchto tónov podobne, ako keď sú bližšie klávesy na klávesnici, tak je väčšia pravdepodobnosť preklepu. Zámena tónov ako taká nesúvisí triviálne s ich vzdialenosťou, ale skôr s funkciou tónu, jeho umiestnení v rámci stupnice a mnohými ďalšími faktormi, ktoré sme bez pomoci (etno)muzikológov nevedeli generalizovať. K definícii tabuľky pre slovenské ľudové piesne, ktorú by sme vedeli využiť, sme sa nedopátrali ani v pomocnej literatúre.

Tento prístup sme týmto neimplementovali, v budúcnosti by však stačilo doplniť do programu jednu jednoduchú metódu do triedy `TermComparator`[5].

### 6.6.4 Problém modulácie

Absolútna reprezentácia je jediná, ktorá je závislá od tóniny piesne. Zároveň máme do činenia s tóninovo normalizovanou databázou tak, ako bola systematicky zapísaná v [2], odkiaľ pochádza väčšina našej databázy. Z týchto dôvodov sme sa rozhodli bližšie neriešiť moduláciu piesní. Ak by sme sa tak rozhodli urobiť, riešili by sme to práve na tomto mieste – podobne, ako pri porovnávaní intervalových reťazcov by sme prečítali reťazec, ktorý sme dostali a vyskúšali všetky rôzne tóniny reťazca. V prípade porovnávania tóninovo nehomogénnych piesní by znova išlo o pridanie jednej metódy do `TermComparator`, ktorá za pomoci `NoteUtils` preiteruje všetkých 12 relevantných tónin a vyberie tú, pri ktorej reťazce dosahujú najvyššiu podobnosť.

### 6.6.5 Riešenie harmonickej variácie

Harmonická variácia prezentovaná v počiatočných kapitolách je veľmi ťažko uchopiteľná algoritmicky. Aj preto ju z našich metód ako jediná rieši reprezentácia pomocou kontúry melódie a aj to len čiastočne. Ako najpresnejšie riešenie harmonickej variácie navrhujeme uloženie najbežnejších dvojíc harmonicky pozmenených melódií do databázy na začiatku algoritmu tak, aby všetky harmonicky podobné melódie boli v jednej triede ekvivalencie. Potom ich algoritmus garantovane zlúči pri všetkých konfiguráciách algoritmu.

Automatizované riešenie tohto problému by existovalo len pri databáze, ktorá obsahuje manuálne zaznačené varianty piesní. Na začiatku by sme jednoducho uložili všetky variantné útržky do tried ekvivalencií.

## 6.7 Váženie tried ekvivalencie

Po rozdelení znotovanej piesne do menších[6] reťazcov a zgrupovaní do tried ekvivalencie nasleduje váženie. K piesni máme z predošlej fázy priradený zoznam tried ekvivalencie a v tejto fáze sa každej triede priradí dôležitosť v rámci piesne. Závisí od počtu výskytov reťazcov z tejto triedy ekvivalencie a súčtu ich počtu v danej piesni,

[5] Detaily na `https://github.com/MichalMizak/SimFolk/blob/master/SimFolk/src/sk/upjs/ics/mmizak/simfolk/core/services/implementations/TermComparator.java`

[6] Pri rozdelení piesne do celej melódie je tento pojem zrejme nepresný, vhodnejšie by bolo hovoriť o menších alebo rovných reťazcoch.

pri istých typoch váh aj od celkového počtu výskytov v databáze. V predošlom algoritme bolo implementovaných viac typov váh. Priblížime si dva hlavné typy.

#### Naivné váhy – *tf*

Pojem *tf* alebo *term frequency* znamená frekvencia výskytu *termu*, v našom prípade triedy ekvivalencie. Každej triede priradíme súčet počtu výskytov všetkých reťazcov v piesni. Dostaneme mapu tried ekvivalencie na počet výskytov – ich váhu.

#### *Tf-idf*

Tento systém priraďuje frázam, ktoré sa v databáze vyskytujú viackrát, nižšiu váhu ako tým, ktoré sú zriedkavé. Bližšie sme ho popísali v bakalárskej práci. Využiť invertovaný index *idf* je vhodné pri veľkých databázach. Keďže oba typy váh dávajú naoko dobré výsledky, ostáva otvorenou otázkou, ktorá z váh je vhodnejšia.

## 6.8 Porovnávanie piesní

### 6.8.1 Formátovanie vektorov

Z predošlých fáz sa až do tejto dostal zoznam piesní s priradenými mapovaniami tried ekvivalencie na ich váhu. Ostáva už len numericky vyčísliť ich podobnosť. Tieto mapovania môžeme chápať ako vektory váh, kde dimenzie sú `id` tried, ku ktorým patria. Vo všeobecnosti tak máme vektory dĺžky $n$, kde $n$ je počet tried v databáze. Nenulové hodnoty váh sú len na tých dimenziách (korešpondujúcim triedam), ktoré sa v piesni nachádzajú. Sú to teda veľmi riedke vektory a porovnávať tieto vektory v takomto formáte nemá zmysel. Pri porovnávaní každých dvoch piesní máme teda dve množiny $A$ a $B$ dimenzií vektorov, ktoré majú nenulové váhy. Musíme zvoliť inklúziu, podľa ktorej budeme vyberať triedy (dimenzie), ktoré chceme brať do úvahy pri porovnávaní. Implementované sú štyri typy inklúzií – zjednotenie, prienik a potom všetky prvky množiny $A$ a všetky typy množiny $B$.

#### Zjednotenie

Táto inklúzia je najobjektívnejšia a generuje najmenej falošne pozitívnych výsledkov. Nech má jedna pieseň formu $ABC$ a druhá formu $ABC$. Nech sú sekcie $AB$ totožné v oboch piesňach, ale sekcia $C$ je rôzna. Označme jednotlivé sekcie $C$ indexami – $C_1$ pre prvú pieseň a $C_2$ pre druhú. Konečné vektory sú potom naformátované vo forme $ABC_1C_2$. Pri vektore prvej piesne je všetkým triedam z $C_2$, ktoré pieseň neobsahuje, priradená nulová váha, pri druhej piesni analogicky pre $C_1$.

#### Prienik

Ak dve piesne obsahujú dve rovnaké frázy, prienik množín ich spoľahlivo odhalí. Prakticky si to na piesňach môžeme predstaviť tak, že ak má pieseň formu $ABC$ a druhá pieseň formu $AB$ a sekcie $B$ sú pre obe piesne rovnaké, výsledná forma bude $B$ a tieto piesne táto inklúzia bude považovať za totožné.

Je zrejmé, že tento prístup prináša veľa falošne pozitívnych výsledkov. Vidieť to napríklad pri piesňach, ktoré majú spoločný len jeden takt a ktoré boli rozdelené do reťazcov dĺžky jedného taktu – veľkosť prieniku je potom 1 a porovnávanie ukáže 100% podobnosť. Tento fakt sme zohľadnili pri agregácii výsledkov, je to však veľmi užitočná inklúzia pri hľadaní spoločných fráz a melodických riadkov, ktoré pri agregovanom výsledku nemusia byť viditeľné.

#### Všetky triedy jednej piesne

Tieto inklúzie sú určené na identifikáciu takých piesní, kde jedna je podmnožinou druhej. Piesne vo formách $AB$ a $ABC$ s rovnakými sekciami $AB$ sú pri inklúzii podľa prvej piesne totožné. Nech je sekcia A totožná a sekcia B rôzna. Pre prvú pieseň ju označme $B_1$, pre druhú $B_2$. Výsledné vektory sú potom naformátované do formy $AB_1$.

Ak by sme porovnávali podľa tried druhej piesne, dostaneme rovnaký výsledok ako pri zjednotení.

### 6.8.2 Porovnávanie vektorov

Finálne vektory dvoch piesní porovnávame pomocou kosínusovej miery, hodnotu ktorej normalizujeme na percentuálnu podobnosť.

## 6.9 Konfigurácia algoritmu

Celý navrhnutý algoritmus je striktne rozdelený do viacerých za sebou nasledujúcich fáz, ktoré sú od seba nezávislé – každá fáza môže fungovať rôznym spôsobom. Tento spôsob je definovaný objektom typu `AlgorithmConfiguration`, ktorý v sebe uchováva všetky voľne nastaviteľné parametre algoritmu. Jeden parameter väčšinou zodpovedá jednej fáze algoritmu, nemusí to tak však striktne byť, komplikovanejšie fázy potrebujú viac vstupných parametrov (samozrejme okrem vstupných piesní). Napríklad fáza rozdeľovania piesní a ich transformácie na reťazec potrebuje špecifikovať reťazcovú reprezentáciu a typ schémy, do ktorej sa pieseň má rozdeliť. Celkový vstup algoritmu sa týmto dá popísať ako množina piesní a jedna konfigurácia algoritmu. Piesne

sa postupne porovnajú každá s každou, ale len jedným smerom – ak sa vykoná porovnanie piesne $P_1$ s piesňou $P_2$, tak sa už nevykoná porovnanie $P_2$ s $P_1$. Je to preto, lebo pre zjednotenie a prienik by sme dostali rovnaký výsledok (keďže sú zrejme symetrické) a ďalšie dve inklúzie sú založené práve na tom, že sa piesne porovnajú raz s ohľadom na množinu tried jednej piesne a druhýkrát na množinu tried druhej piesne.

# 7 Agregačný algoritmus

## 7.1 Generovanie konfigurácií algoritmu

Algoritmus, o ktorom sme doposiaľ celý čas hovorili je navrhnutý tak, že každá konfigurácia nájde iný typ podobnosti – podobnosti menších fráz, rytmické podobnosti, kontúry a podobne. Preto ho môžeme nazvať iba akýmsi subalgoritmom. Náš prístup je tieto samostatné rôznorodé podobnosti zhrnúť do jednej aproximačnej hodnoty podobnosti navrhnutím algoritmu, ktorý výsledky tohto subalgoritmu agreguje. Tento agregačný algoritmus sme navrhli tak, že na vstup dostane len `.xml` súbory piesní, ktoré sa majú porovnávať a následne sa všetky zmysluplné konfigurácie *vygenerujú*. Generované sú v triede `MusicAlgorithmConfigurationGenerator` pomocou *backtrack* metódy a *builder pattern-u* – generujeme všetky kombinácie všetkých implementácií jednotlivých fáz.

Po vygenerovaní konfigurácií sa algoritmus v cykle spúšťa pre každú konfiguráciu osobitne a každý vyrátaný čiastkový výsledok sa pošle agregačnému rozhraniu na spracovanie.

### Vylepšenia rýchlosti

Algoritmus sme navrhli tak, že po vyrátaní sa každý výsledok posiela na spracovanie do radu úloh iného vlákna. Týmto vieme paralelne spracovávať jeden výsledok a zároveň rátať ďalší výsledok s inou konfiguráciou algoritmu, čo značne zrýchli celý výpočet.

V prípade veľmi veľkého datasetu je možné veľmi jednoducho rozšíriť program o *REST API*, ktoré pošle výsledky na spracovanie na iný server a tým odľahčí stroj, ktorý ráta jednotlivé subalgoritmy. Zvažovali sme aj paralelizovanie rátania jednotlivých konfigurácií, tam sa však vyskytol problém s paralelnými prístupmi k databáze a vzájomnými konfliktmi niektorých konfigurácií. Čo by ale bolo možné, je rozdeliť množinu vygenerovaných konfigurácií na viac častí a spúšťať každú časť na inom stroji a výsledky na spracovanie posielať na jeden server. Rýchlosť algoritmu však nebola priorita a kvôli tomu sme implementáciu vyradili z cieľov práce.

Ako posledné vylepšienie sme implementovali vyrátanie všetkých inklúzií pre dvojicu piesní v rámci jedného behu subalgoritmu naraz. Dá sa to preto, pretože existujú štyri konfigurácie, ktoré majú všetky fázy totožné až na poslednú – formátovanie a následné porovnávanie vektorov. Štyri preto, pretože sme implementovali štyri rôzne inklúzie a jeden porovnávací algoritmus. Zrýchlenie je teda štvornásobné.

## 7.2 Dátová štruktúra výsledku

Pri zvažovaní formy spracovania výsledkov sme najprv museli identifikovať dátovú štruktúru, ktorá najlepšie odráža charakter výsledkov. Každý beh subalgoritmu vráti zobrazenie dvojíc piesní do množiny $[0, 100]$[1]. Takáto funkcia sa dá veľmi vhodne interpretovať ohodnoteným grafom, kde vrcholy sú piesne a ohodnotené hrany percentuálne podobnosti medzi nimi. Grafová štruktúra je zároveň veľmi názorne vizualizovateľná a vhodná na analýzu dát pomocou algoritmov na analýzu grafových dát (*graph data science* algoritmov).

## 7.3 Grafová databáza *Neo4j*

Výsledok algoritmu potrebujeme, prirodzene, niekam uložiť. Pri prieskume technológií, ktoré umožňujú ukladanie grafov, sme prezreli viacero rebríčkov popularity[2], kde medzi najpopulárnejšie patrí jednoznačne grafová databáza *Neo4j*. Grafové databázy sú vhodné na ukladanie takých dát, kde sa kladie dôraz na vzťahy medzi objektmi. To je aj náš prípad.

### Vizualizácia

Táto technológia nám umožní aj jednoduchú, ale užitočnú vizualizáciu grafu pomocou nástroja *Neo4j browser*. S databázou sa okrem iných metód dá priamo komunikovať pomocou jazyka *Cypher*. Syntax jazyka je príjemná a expresívna a teda, aj keď sme s týmto jazykom nikdy predtým nepracovali, dopytovanie je jednoduché.

### Technologické detaily

Aby sme mohli používať databázu *Neo4j*, nainštalovali sme si lokálny server, na ktorom databáza beží. Komunikujeme s ním vďaka špeciálnemu binárnemu *Bolt* protokolu, ktorý poskytuje Java knižnica *Spring Data Neo4j*. Tento prístup nám umožní

[1] Najčastejšie je dvojiciam, prirodzene, priradená hodnota 0, ktorá naznačuje úplnú rozdielnosť

[2] Napríklad rebríček `https://www.c-sharpcorner.com/article/most-popular-graph-databases/`

meniť databázu z lokálnej na vzdialenú bez potreby zmeny kódu – stačí zmeniť adresu servera, používateľské meno a heslo k databáze.

## 7.4 Ukladanie čiastkových výsledkov

Čiastkové výsledky po jednom prichádzajú do rozhrania `MelodyResultAggregatorService`. To najprv uloží tento výsledok do databázy ako nový graf. V praxi to znamená, že databáza obsahuje veľkú množinu grafov, ktoré spolu logicky nesúvisia – sú z „rôznych svetov“, korešpondujú rôznym konfiguráciám. Počet takýchto logicky oddelených grafov je presne počet spracovaných konfigurácií, ktoré boli vygenerované na začiatku algoritmu. Grafy od seba vieme odlíšiť podľa popisu a vlastností ich hrán medzi jednotlivými vrcholmi. Hrany totiž nesú okrem percentuálnej hodnoty podobnosti aj informácie o konfigurácii, pomocou ktorej boli vyrátané (*Neo4j* podporuje ukladanie grafov typu *property graph*).

Jazyk *Cypher* podobne ako jazyk *SQL* podporuje syntax `WHERE` podmienky. Pre dopyt na získanie grafu korešpondujúcemu konkrétnej konfigurácii stačí do podmienky pridať vlastnosti (konfiguračné parametre fáz subalgoritmu), ktoré nás zaujímajú.

## 7.5 Agregačný graf

Po uložení dielčieho výsledku ho zarátavame do celkového aproximačného výsledku nasledovne: každej konfigurácii je pevne priradená dôležitosť – váha z intervalu $[0, 1]$. Následne sa pre každú dvojicu piesní ráta vážený priemer podobností. Podobnosť má v celkovom súčte takú váhu ako konfigurácia, pomocou ktorej bola vyrátaná. Konfigurácie sú ohodnocované osobitne. Prioritne sme týmto spôsobom chceli priradiť nižšiu dôležitosť konfiguráciám s toleranciou – čím vyššia tolerancia, tým nižšia dôležitosť. Následne sme znižovali dôležitosť konfiguráciám, ktoré rátali podobnosť pre prienik vektorov a mierne sme museli kvôli veľkému množstvu falošne pozitívnych výsledkov znížiť váhu rytmickej reprezentácii.

Po tomto procese je graf uložený do databázy s vlastnosťou `aggregationResult` nastavenou na `true`. Vďaka tejto vlastnosti vieme výsledný graf jednoducho dopytovať z databázy ako podgraf jedného obrovského grafu obsahujúceho výsledky všetkých behov algoritmu so všetkými konfiguráciami.

Za kľúčový považujeme fakt, že tento agregačný graf je dostupný už po spracovaní prvého čiastkového výsledku a potom sa už len aktualizujú jeho hodnoty. Na úplné riešenie si síce musíme počkať do konca behu celého algoritmu, ale čiastkové riešenie je užívateľovi dostupné už po vyrátaní prvej konfigurácie. Zvyčajne sa podľa našich

pozorovaní výsledky ustália po vyrátaní polovice konfigurácií.

## 7.6 Porovnávanie metadát

Pri analýze existujúcich riešení sme pozorovali, že pri porovnávaní slovenských ľudových piesní je kladený veľký dôraz na ich metadáta – vlastnosti ako región, hudobná forma, centrálny tón a podobne.

### 7.6.1 Návrh riešenia porovnávania metadát

Navrhli sme jednoducho implementovateľné riešenie. Mimo výsledkov porovnávania piesní pomocou navrhnutého subalgoritmu by sme implementovali ďalší algoritmus. Tento algoritmus by na vstupe dostal objekt obsahujúci všetky dostupné metadáta (pri neúplnej databáze len také, ktoré majú obe piesne vyplnené) o piesňach. Tieto objekty by sa navzájom porovnali tak, že by sa dal do pomeru počet všetkých a počet zhodných atribútov a previedol sa na percentá. Prípadne by sa mohla priradiť rôzna dôležitosť jednotlivým atribútom podľa odbornej evaluácie dôležitosti (napríklad folklórny región piesne je intuitívne menej dôležitý ako presná lokalita, kde bola pieseň zaznamenaná).

Výsledkom porovnania všetkých piesní týmto algoritmom navzájom by bol úplný graf podobnosti. Tento graf by sme agregovali podobne ako ostatné čiastkové výsledky. Malo by však zmysel dať mu väčšiu dôležitosť, ako výsledku vyššie navrhnutého subalgoritmu pre jednu konfiguráciu. Namiesto hodnoty z intervalu $[0, 1]$ sa odhadom ako vhodné javí číslo 100. Číslo by sa ešte mohlo špecifikovať podľa dôležitosti jednotlivých parametrov. Toto porovnanie metadát by bolo významným prínosom do nášho algoritmu, s našou databázou je však, žiaľ, nerealizovateľné.

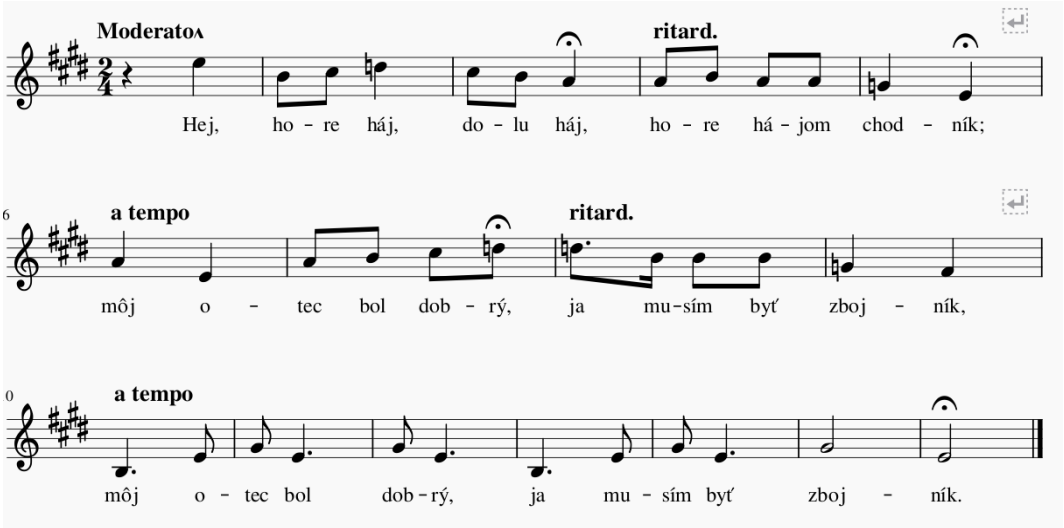
# 8 Výsledky algoritmu

## 8.1 Výsledky prototypu algoritmu

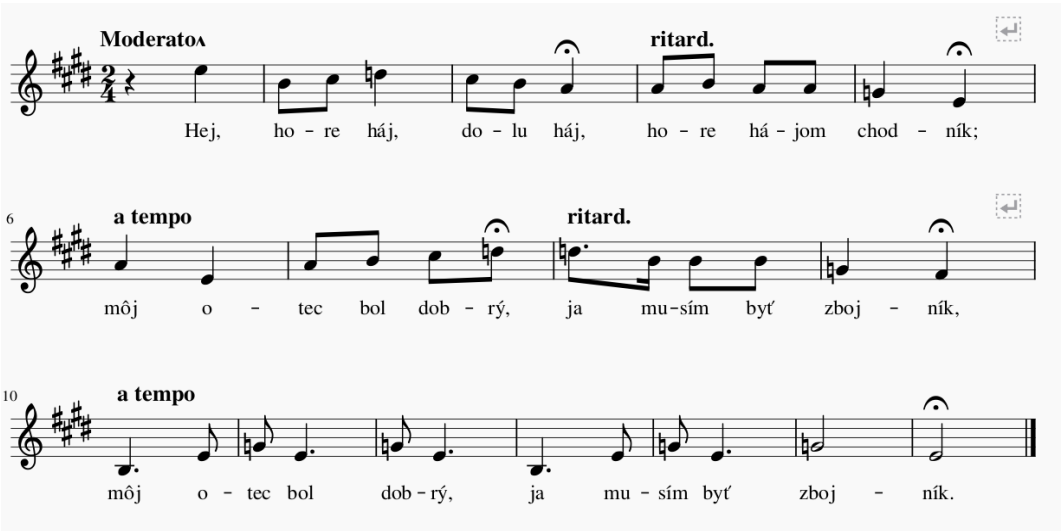
Po spustení algoritmu v prvej implementácii sme prešli všetky dostupné výsledky a našli najreprezentatívnejšiu dvojicu spomedzi všetkých porovnávaných piesní. Táto dvojica je odvodená z jednej piesne *Hore háj, dolu háj* a jasne ukazuje fenomén variantnosti. Dosiahnutá podobnosť bola 95%. Výsledok bol dosiahnutý vďaka nasledovnej konfigurácii algoritmu:

| fáza | stratégia fázy |
|---|---|
| reťazcová reprezentácia | ABSOLUTE |
| rozdeľovanie melódií | MEASURE_NGRAM |
| porovnávanie melodických útržkov | LEVENSHTEIN_DISTANCE |
| tolerancia | MEDIUM |
| váženie tried ekvivalencie | TF · IDF |
| formovanie vektorov do inklúzií | A forma |
| porovnávanie vektorov | COS_SIMILARITY |

Tabuľka 7: Konfigurácia prototypného behu algoritmu




Obr. 16: Hore háj, dolu háj verzia 1




Obr. 17: Hore háj, dolu háj verzia 2

Keď sa na piesne pozrieme bližšie, vidíme, že sa odlišujú len v poslednom riadku. Variantnosť je zrejme spôsobená posledným taktom v druhom riadku piesne, kde sa z durovej tóniny prejde do molovej. Vo verzii 1 spevák (resp. človek, ktorý bol zaznamenávaný pri zápise tejto piesne) v poslednom riadku prešiel naspäť do durovej tóniny, vo verzii 2 bola zaznamenaná molová verzia.

## 8.2 Analýza falošne pozitívnych výsledkov a evaluácia konfigurácií

Pri algoritme sme potrebovali identifikovať konfigurácie, ktoré vracajú najviac falošne pozitívnych výsledkov. Pri spustení algoritmu na 250 piesňach z datasetu sme v databáze mali takmer 35 955 rôznych uzlov grafov a 907 122 hrán, z toho 897751 hrán a 35707 uzlov je z behu subalgoritmu pre jednu konfiguráciu. Jedna hrana reprezentuje to, že dva uzly majú podobnosť väčšiu ako 50%. Je to tak preto, lebo ak by sme chceli ukladať všetky, aj nulové podobnosti, pre 250 piesní a 256 konfigurácií by sme dostali $250 \cdot 256 = 64000$ uzlov a $250^2 \cdot 256$ hrán medzi nimi. Ak teda pieseň v čiastkovom výsledku nie je podobná so žiadnou piesňou na viac ako 50%, tak sa nenachádza ako uzol v grafe reprezentujúcom čiastkový výsledok. Do celkového výsledku sa však zaráta každá, aj nulová podobnosť medzi piesňami.

Pri implementácii sme tušili, že najväčší počet falošných výsledkov môže nastať pri inklúzii vektorov a nastavení tolerancie. Preto sme naformulovali nasledovné *CYPHER* dopyty:

```
MATCH ()-[r:IS_SIMILAR]->()
WHERE NOT r.aggregationResult
RETURN r.tolerance as Tolerancia, r.vectorInclusion as Inklúzia,
COUNT(*) AS Počet
ORDER BY Počet DESC
```

```
MATCH ()-[r:IS_SIMILAR]->()
WHERE NOT r.aggregationResult
RETURN r.tolerance as Tolerancia, COUNT(*) AS Počet
ORDER BY Počet DESC
```

```
MATCH ()-[r:IS_SIMILAR]->()
WHERE NOT r.aggregationResult
RETURN r.vectorInclusion as Inklúzia, COUNT(*) AS Počet
ORDER BY Počet DESC
```

Tie vrátili nasledovné tabuľky:

| tolerancia | inklúzia | počet |
|---|---|---|
| HIGH | INTERSECTION | 125106 |
| MEDIUM | INTERSECTION | 103325 |
| LOW | INTERSECTION | 88867 |
| NONE | INTERSECTION | 87164 |
| HIGH | B | 55411 |
| HIGH | A | 54299 |
| MEDIUM | B | 48182 |
| MEDIUM | A | 46561 |
| LOW | B | 44655 |
| NONE | B | 44069 |
| LOW | A | 42989 |
| NONE | A | 42522 |
| HIGH | UNIFICATION | 31087 |
| MEDIUM | UNIFICATION | 28312 |
| LOW | UNIFICATION | 27709 |
| NONE | UNIFICATION | 27493 |

Tabuľka 8: Tabuľka počtov podobností väčších ako 50% pre rôzne tolerancie a inklúzie

| tolerancia | počet |
|---|---|
| HIGH | 265903 |
| MEDIUM | 226380 |
| LOW | 204220 |
| NONE | 201248 |

Tabuľka 9: Počty podobností väčších ako 50% pre rôzne tolerancie

| inklúzia | počet |
|---|---|
| INTERSECTION | 404462 |
| B | 192317 |
| A | 186371 |
| UNIFICATION | 114601 |

Tabuľka 10: Tabuľka počtov podobností väčších ako 50% pre rôzne inklúzie

Tabuľky sú zoradené podľa posledného stĺpca. Podľa očakávaní má inklúzia väčší dopad na výsledok ako tolerancia. Rozdiel v počtoch hrán pri najvyššej a najnižšej tolerancii sú v prepočte na počet uzlov len 2 hrany – každý uzol bude incidovať s dvoma ďalšími vrcholmi. Pri inkúziách to je však dramatickejšie – rozdiel je takmer 300000, čo je v prepočte takmer 9 hrán. Najviac podobností vracia inklúzia prieniku, potom inklúzie podľa ľavého alebo pravého vektora (`A` alebo `B`) a najmenej podobností vracia zjednotenie. Inklúzii prieniku sme teda priradili podstatne nižšiu váhu pri agregácii, ako ostatným inklúziám. Toleranciám sme ponechali rovnaké váhy.

Obrovský rozdiel je aj pri type rozdeľovania piesní. Tam sme dostali tabuľku

| schéma | počet |
|---|---|
| takt | 747890 |
| n-gram 4 taktov | 145441 |
| celá pieseň | 4420 |

Tabuľka 11: Tabuľka podľa rozdeľovania piesní do schém

Tu sme tiež dostali obrovský rozdiel. Zohľadnili sme aj ten a menšiu dôležitosť sme priadili aj rozdeľovaniu do schémy taktu, aj celej piesne.

Po tomto pokuse sme ešte pridali intervalovú fuzzifikáciu a upravili kontúrovú reprezentáciu, výsledky ostali podobné. Váhy sme priradili podľa takejto tabuľky:

| inklúzia vektorov | schéma | reťazcová reprezentácia | váha |
|---|---|---|---|
| prienik | takt | ľubovoľná | 0,25 |
| prienik | ľubovoľná | rytmická | 0,25 |
| prienik | ľubovoľná | kontúrová | 0,25 |
| prienik | ľubovoľná | ľubovoľná | 0,5 |
| ľubovoľná | takt | rytmická | 0,5 |
| ľubovoľná | takt | ľubovoľná | 0,75 |

Tabuľka 12: Tabuľka podľa rozdeľovania piesní do schém

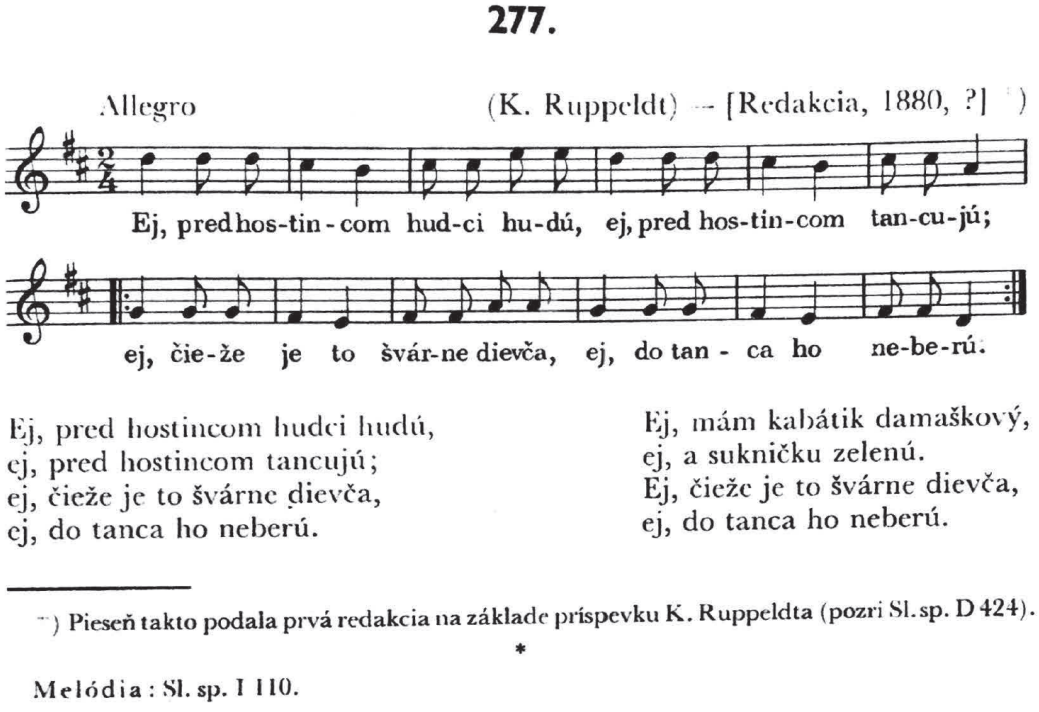
Inak je konfigurácii priradená váha 1. Výsledky behov algoritmov sa potom pre každú dvojicu piesní agregujú do jednej hodnoty váženým priemerom, kde hodnota je percentuálna podobnosť medzi piesňami a váha je hodnota evaluácie konfigurácie algoritmu podľa tabuľky 12.

## 8.3 Výsledky finálneho algoritmu

Algoritmus sme púšťali pre všetkých 517 piesní, ktoré sme mali k dispozícii. V databáze sme potom mali celkovo 102007 uzlov a 3915067 hrán medzi nimi. Z toho 1699 hrán medzi 381 uzlami patrí agregovanému grafu. Dostávame teda, že $517 - 381 = 136$ piesní nie je podobných so žiadnou piesňou databázy na viac, ako 60%.

Zaujímavá je aj distribúcia percentuálnych podobností – hrán s ohodnotením 60 až 70 percent bolo v agregovanom grafe 1441 medzi 353 uzlami, hrán s ohodnotením vyšším ako 70 percent bolo 258 medzi 156 uzlami.

Prvý konkrétny výsledok, ktorý uvedieme, je podobnosť nasledujúcich dvoch piesní:



**277.**

Allegro (K. Ruppeldt) – [Redakcia, 1880, ?] *)

Ej, pred hos-tin-com hud-ci hu-dú, ej, pred hos-tin-com tan-cu-jú;

ej, čie-že je to švár-ne dievča, ej, do tan - ca ho ne-be-rú.

Ej, pred hostincom hudci hudú,
ej, pred hostincom tancujú;
ej, čieže je to švárne dievča,
ej, do tanca ho neberú.

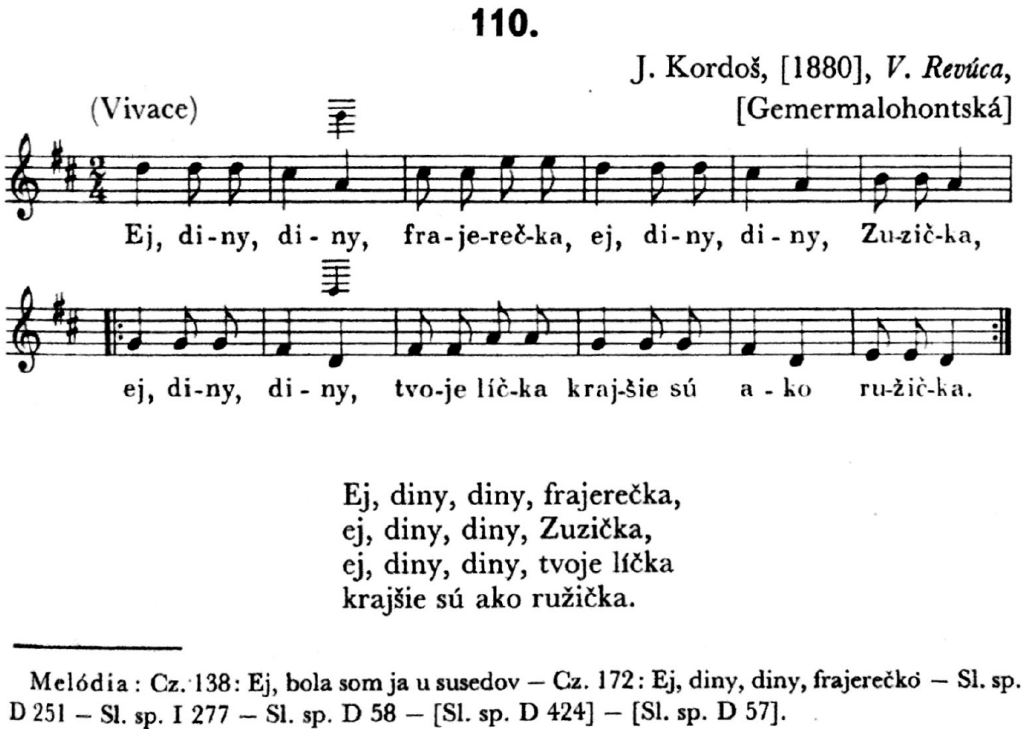
Ej, mám kabátik damaškový,
ej, a sukničku zelenú.
Ej, čieže je to švárne dievča,
ej, do tanca ho neberú.

*) Pieseň takto podala prvá redakcia na základe príspevku K. Ruppeldta (pozri Sl. sp. D 424).

*

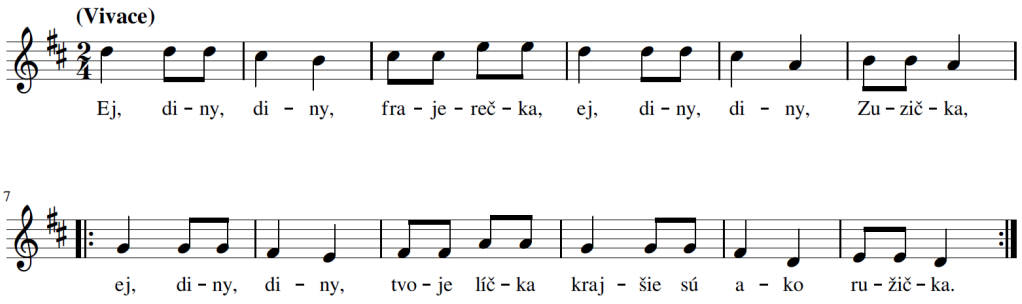
Melódia: Sl. sp. I 110.

Obr. 18: *Ej, pred hostincom hudci hudú*



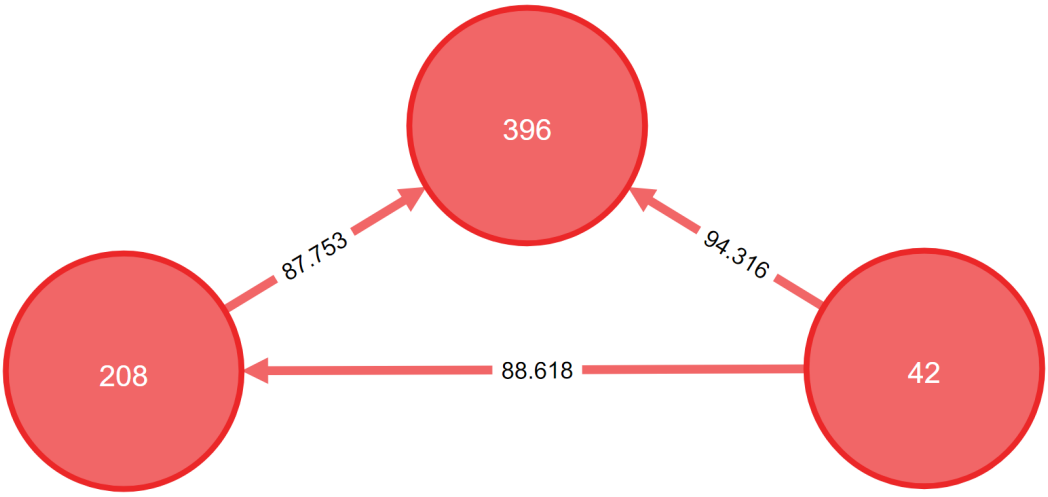

Obr. 19: *Ej, diny, diny, frajerečka*

Obrázky 22 a 20 sú skeny strán 323 a 203 zo zbierky [2]. Pri zápise piesne *Ej, diny, diny, frajerečka* si môžeme všimnúť aj zaznačenie variantu (tóny nad notovou osnovou). Tento variant sa v databáze nachádza tiež:




Obr. 20: Variant piesne *Ej, diny, diny, frajerečka*

Algoritmus dokázal identifikovať vysokú podobnosť medzi všetkými troma piesňami. Na obrázku 21 vidíme časť grafu agregovaného výsledku, kde je pieseň *Ej, diny, diny, frajerečka* označená číslom 42, jej variant číslom 369 a pieseň *Ej, pred hostincom hudci hudú* je označená číslom 208. Ohodnotenie hrán je percentuálna podobnosť medzi nimi.

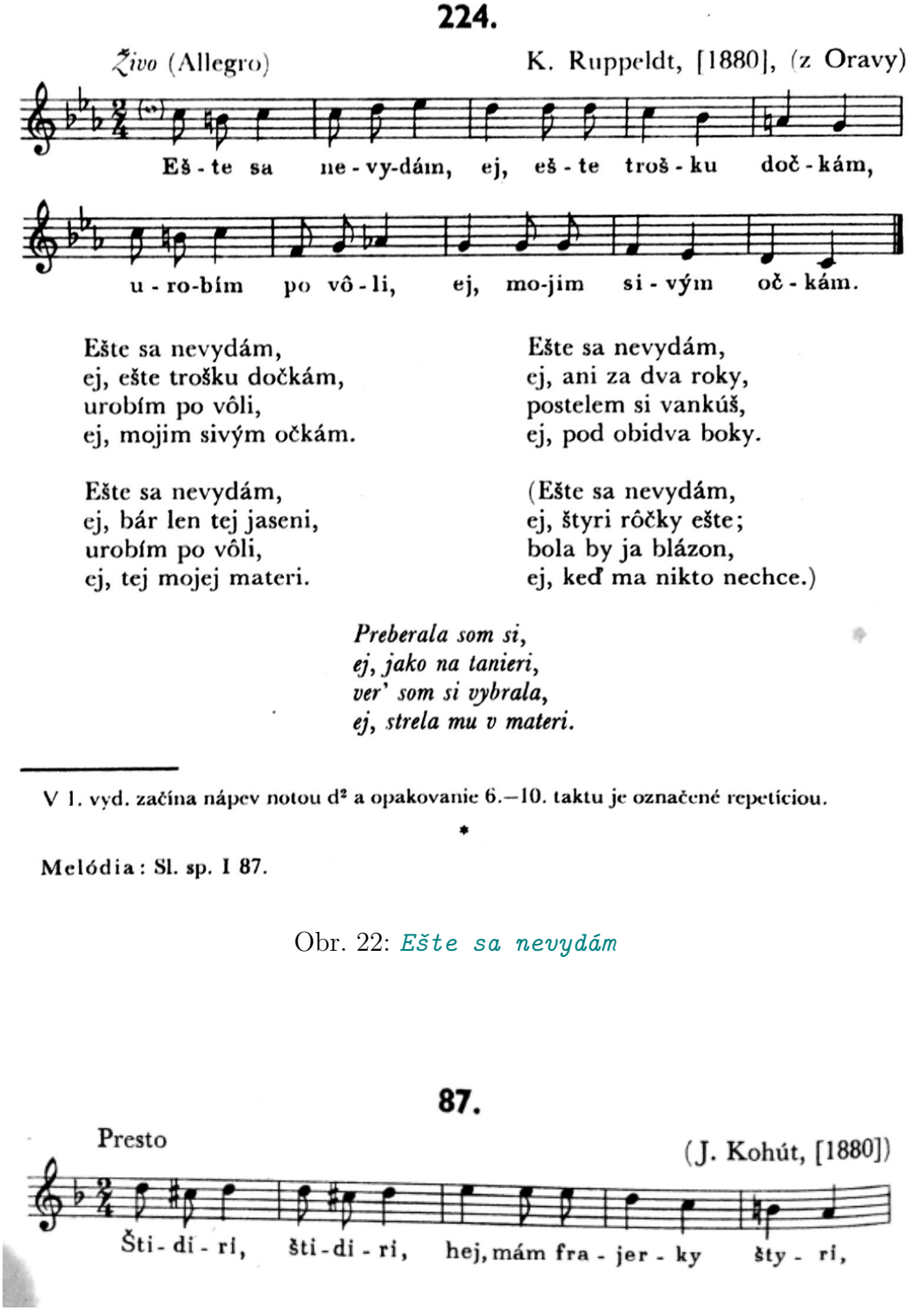



Obr. 21: Výsledok algoritmu pre melodickú rodinu

Tieto piesne môžeme určite zaradiť do jednej melodickej rodiny a, ako vidíme, algoritmus správne identifikoval ich vysokú podobnosť.

### 8.3.1 Modulované piesne

Pozrieme sa aj na fenomén modulovaných variantov:



**224.**

*Živo* (Allegro) K. Ruppeldt, [1880], (z Oravy)

Eš - te sa ne - vy-dám, ej, eš - te troš - ku doč - kám,
u - ro-bím po vô - li, ej, mo-jim si - vým oč - kám.

Ešte sa nevydám,
ej, ešte trošku dočkám,
urobím po vôli,
ej, mojim sivým očkám.

Ešte sa nevydám,
ej, ani za dva roky,
postelem si vankúš,
ej, pod obidva boky.

Ešte sa nevydám,
ej, bár len tej jaseni,
urobím po vôli,
ej, tej mojej materi.

(Ešte sa nevydám,
ej, štyri rôčky ešte;
bola by ja blázon,
ej, keď ma nikto nechce.)

*Preberala som si,*
*ej, jako na tanieri,*
*ver' som si vybrala,*
*ej, strela mu v materi.*

V 1. vyd. začína nápev notou $d^2$ a opakovanie 6.—10. taktu je označené repetíciou.

*

Melódia: Sl. sp. I 87.

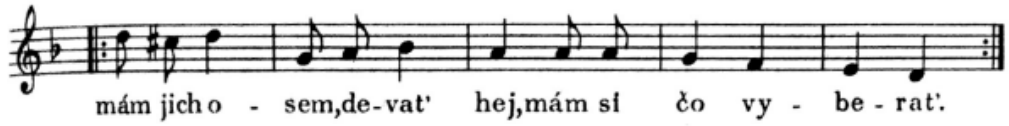
Obr. 22: *Ešte sa nevydám*



**87.**

Presto (J. Kohút, [1880])

Šti - di - ri, šti - di - ri, hej, mám fra - jer - ky šty - ri,




Štidiri, štidiri,
hej, mám frajerky štyri,
mám jich osem, devať,
hej, má si čo vyberať.

Frajerečky štyri,
hej, prečo ste sa bili?
Pre teba, šuhajko,
hej, keď sme ťa ľúbili.

Frajerečky štyri,
hej, pre mňa sa nebite,
veď vy mojou ženou,
hej, žiadna nebudete.

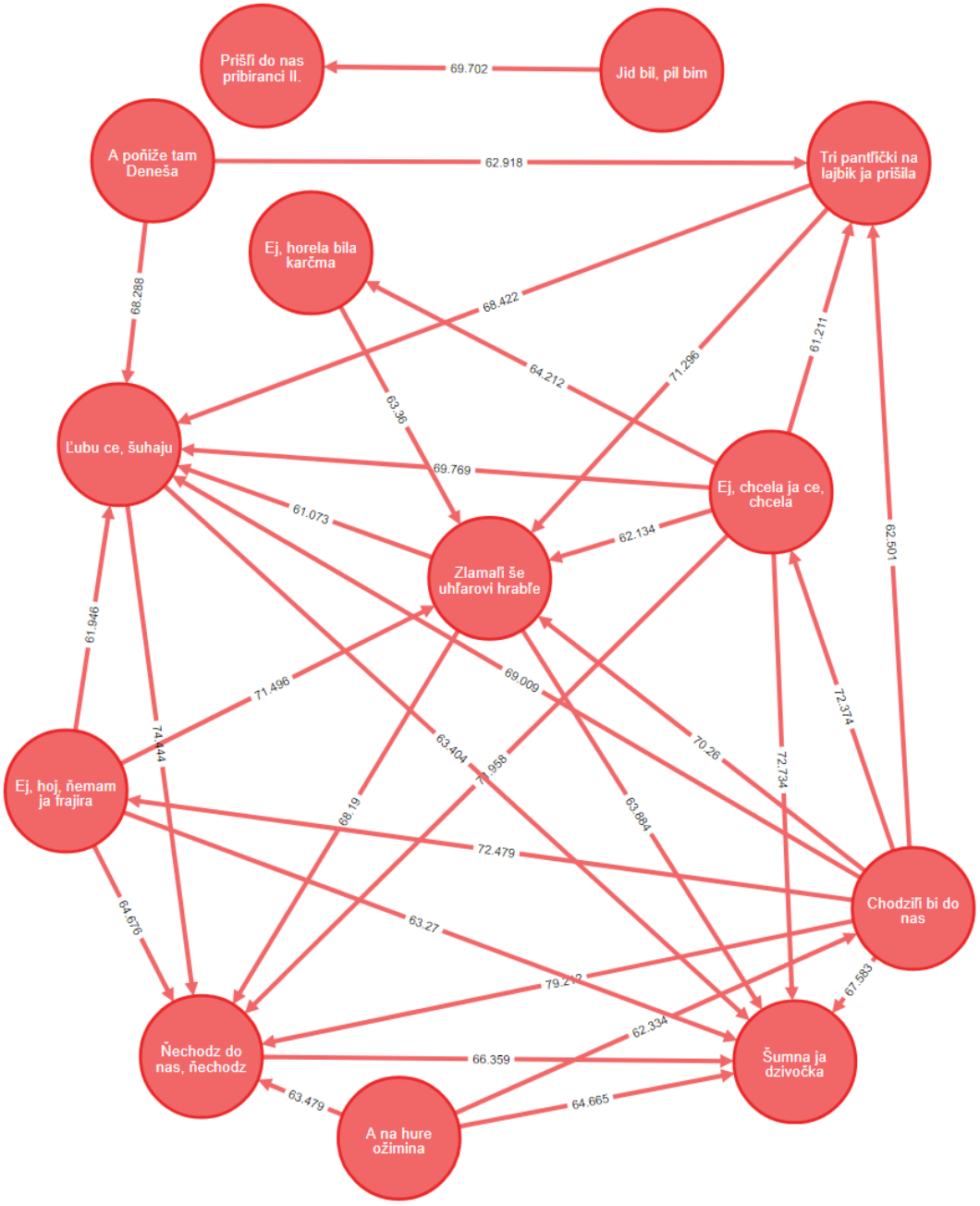
Melódia: Cz. 189: Šuhajko luterán, ej, za teba vuoľu mám – Cz. 95: Henok na tej hrobli, ej, tam sa farár modlí – Cz. 111: Šuhajko biely vták, ej, zaleteu za Frajšták – Cz. 248: Bola som ja, bola, ej, v zelenom hájičku – Sl. sp. I 224 – Sl. sp. D 93 – Sl. sp. D 2097 – Sl. sp. D 446 – [Sl. sp. III 423].

Obr. 23: *Štidiri, štidiri, hej, mám frajerky štyri*

Obrázky sú znova skeny zo zbierky [2] zo strán 285 a 188. Pre tieto dva varianty, ktoré sú navyše v rôznych tóninách, algoritmus vyhodnotil podobnosť na 88%. To len dokazuje, že aj pri piesňach, ktoré nie sú tóninovo normalizované, vylepšiť identifikáciu modulovaných piesní porovnávaním piesní vo všetkých tóninách pri absolútnej reprezentácii nebolo nutné.

### 8.3.2 Piesne repertoáru FS Zemplín

Zaujímali nás aj výsledky porovnávania piesní z repertoáru FS Zemplín. Medzi týmito piesňami sa nachádzajú piesne z rôznych lokalít, napríklad z dedín Dvorianky, Veľké Zalužice alebo Dlhé nad Cirochou.




Obr. 24: Agregačný graf zemplínskych melódií

Veľmi zaujímavé pozorovanie je, že piesne sú len v dvoch rôznych komponentoch v grafe. Zároveň sú v grafe veľmi pevne prepojené, čo naznačuje silnú homogenitu repertoáru súboru. Na druhej strane, keď sme dopytovali všetky piesne, ktoré vo výslednom grafe incidujú s piesňami tohto repertoáru, vo výsledku bolo vrátane zemplínskych 104 piesní. V celkovom grafe boli stupne uzlov zemplínskych piesní boli medzi 1 a 30.

Ďalej sme skúmali súčet podobností vo všetkých konfiguráciách pre všetky tieto piesne. Za zaujímavý považujeme fakt, že pieseň *Zarmucil ši, ti, beťaru* mala spomedzi všetkých 517 piesní tretí a pieseň *Šumna ja dzivočka* štvrtý najmenší súčet

všetkých podobností pre všetky grafy čiastkových výsledkov. Pieseň *Zarmucil si, ti, beťaru* sa do agregačného grafu vôbec nedostala, čo hovorí o nízkej podobnosti vrámci piesní a nepriamo svedčí o jej originalite.

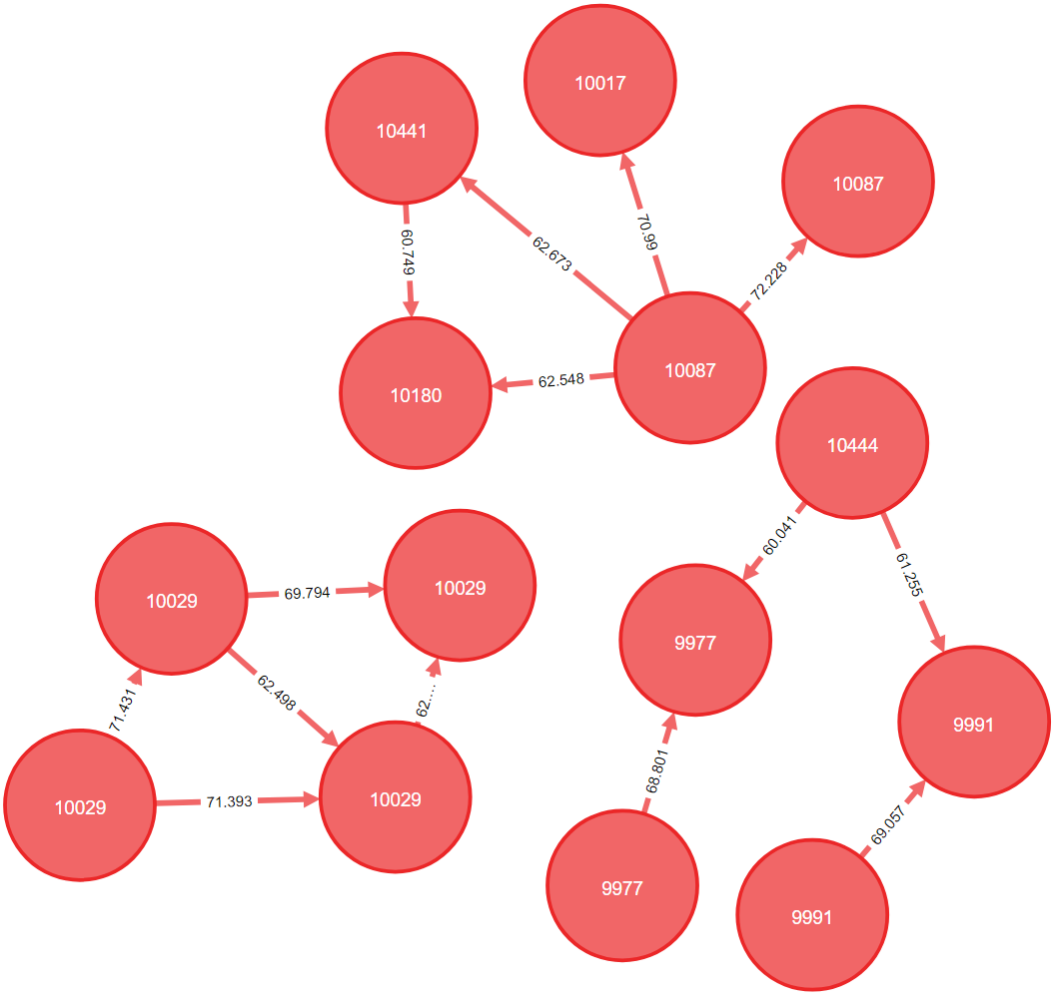
## 8.4 Analýza výsledkov

*Neo4j* sme si zvolili aj kvôli faktu, že v sebe nesie množstvo efektívne implementovaných grafových algoritmov. Dajú sa skúmať cesty medzi vrcholmi, počet komponentov výsledného grafu, stabilita siete. Dokonca sú tam aj algoritmy, ktoré analyzujú podobnosť podgrafov alebo vrcholov podľa susedov a ich vlastností. To nám umožňuje študovať napríklad podobnosť v rámci výsledkov subalgoritmu pre rôzne konfigurácie.

### 8.4.1 Aproximácia melodických rodín

Na záver sme na analýzu našich dát využili známy algoritmus *label propagation*. Tento algoritmus bol vyvinutý na to, aby označil rovnakou hodnotou uzly, ktoré by mohli tvoriť komunitu[1]. Preto môže slúžiť na dobrú aproximáciu melodických rodín. Algoritmus je na rozdiel od algoritmu hľadania komponentov jemnejší a dokáže na základe váhy, v našom prípade podobnosti, rozdeliť uzly do menších melodických rodín. Napríklad všetky piesne melodickej rodiny piesne *Ej, diny, diny, frajerečka* označil algoritmus rovnakým číslom, čo dokazuje úspešnosť algoritmu.

[1] Viac implementačných detailov algoritmu v dokumentácii databázy *Neo4j*.




Obr. 25: *Label propagation* na agregačnom grafe

# Záver

V závere bakalárskej práce „Hľadanie podobností v textoch ľudových piesní“ bolo zadefinovaných viacero cieľov do budúcna. Jedným z nich bola štatistická analýza efektivity jednotlivých konfigurácií algoritmu. Tento cieľ sme v rozšírení tejto práce splnili a analyzovali konfigurácie dostatočne na to, aby sme mohli implementovať taký agregačný algoritmus všetkých konfigurácií, ktorý vhodne odráža realitu podobností medzi objektami.

Ďalším cieľom do budúcna bol návrh a implementácia algoritmu na hľadanie podobností v melódiách ľudových piesní. Tento cieľ sme takisto splnili. Navyše sme výsledky uložili do grafovej databázy *Neo4j*, analyzovali ich pomocou zabudovaných grafových algoritmov jej knižníc a pomocou technológie *Neo4j Browser* tieto výsledky vizualizovali. Naprogramovaný algoritmus je použiteľný ako pre odborných (etno)muzikológov, tak aj pre hudobníkov, folkloristov a hudobných nadšencov. Ukázali sme, že nepriamo sa ním dá analyzovať napríklad aj miera originality piesní alebo homogénnosť repertoáru.

Keďže by sa dal vylepšovať donekonečna, existujú rôzne možnosti rozširovania tohto algoritmu. V práci sme ich navrhli viacero – distribuovať a paralelizovať algoritmus, ukladanie variantných fráz do tried ekvivalencií alebo automatizovaná analýza metadát piesní. Mnoho z týchto prístupov je použiteľných len za predpokladu širokej databázy piesní s metadátami a manuálnymi analýzami muzikológov. Optimalizácie algoritmu je možné urobiť pri potrebe vylepšenia rýchlosti algoritmu ihneď bez ďalších predpokladov.

Veríme, že vypracovaním tejto práce sme prehĺbili prienik muzikológie a informatiky na Slovensku a dopomohli týmto analýze nášho kultúrneho dedičstva, ktorého nespochybniteľne dôležitou časťou sú ľudové piesne.

# Zoznam použitej literatúry


[1] Elschek, O., and Elscheková, A. *Úvod do štúdia ľudovej hudby.* Hudobné centrum, Bratislava, 2008.

[2] Galko, L. *Slovenské spevy. 1. diel.* Opus, Bratislava, 1972.

[3] Janošcová, R. Počítačová podpora em, 08 2015.

[4] Kresánek, J. *Slovenská ľudová pieseň zo stanoviska hudobného.* Hudobné centrum, Bratislava, 1997. ISBN: 80-88884-14-4.

[5] Lariš, B. *Tonálna a rytmická charakteristika hudobného folklóru Kysúc.* PhD thesis, Univerzita Palackého v Olomouci, Pedagogická fakulta, Olomouc, 2010.

[6] Müllensiefen, D., and Frieler, K. Cognitive adequacy in the measurement of melodic similarity: Algorithmic vs. human judgments. *Music Query: Methods, Models, and User Studies Computing in Musicology, 13* (2003).

[7] Ľubomír Chalupka. Modelovanie tonálnej a formovej analýzy ľudových piesní zo záhoria na počítači msp 2a (spomienka na 70. roky uplynulého storočia). *K otázkam typologie tradiční hudby, Sborník referátu ze stejnojmenné mezinárodní konference konané 21.9 – 22.9.2016 v Praze* (2016), 61 – 72.

[8] van Kranenburg, P., Garbers, J., Volk, A., Wiering, F., Grijp, L., and Veltkamp, R. Collaboration perspectives for folk song research and music information retrieval: The indispensable role of computational musicology. *Journal of Interdisciplinary Music Studies* (2009).

[9] Volk, A., Van Kranenburg, P., Garbers, J., Wiering, F., Veltkamp, R., and Grijp, L. The study of melodic similarity using manual annotation and melody feature sets.