Sveučilište Jurja Dobrile

# Algoritamsko skladanje glazbe putem funkcijskog programiranja

Izradili: Luka Grubeša, Lara Martinčević, Marko Franković

U Puli,2024.

Sadržaj:

## Uvod:

Funkcijsko programiranje glazbe u Haskellu nekonvencionalan je pristup stvaranju glazbe. Ovim projektom istražili smo neke od mogućih funkcionalnosti Haskella i njegovi library-ja te ih primijenili na glazbenu strukturu.

## Korišteni algoritmi u svrsi obrade glazbe:

- ### Markovljev Lanac(Markov Chain):

  Markovljev lanac je algoritam koji funkcionira na temelju konačnog slijeda elemenata u našem slučaju glazbenih, gdje svaki element sadrži informaciju o mogućem prijelazu u sljedeće stanje te pripadajući postotak vjerojatnosti tog prijelaza. Kroz ovaj mehanizam, algoritam predviđa sljedeće note na temelju prethodnih.

- ### Celular Automata:

  Cellular Automata je algoritam koji predstavlja koncept u kojem se glazbena struktura stvara unutar dvodimenzionalnog grida, gdje svaka stanica u rešetki ima određeno stanje koje evoluira prema definiranim pravilima. U ovom kontekstu, "#" može predstavljati ton, dok "." označava pauzu. Kroz iterativno primjenu pravila na početnom stanju generiraju se glazbeni uzorci.

Predikcija sljedećih nota putem Markovljevog lanca pruža temeljnu strukturu, dok Cellular Automata dodaje slojeve varijacije i kompleksnosti. Kombinirajući ove pristupe, stvaratelji glazbe mogu kreirati kompozicije koje istražuju različite aspekte glazbenog stvaralaštva, od tradicionalnih do eksperimentalnih, nudeći inspirativne mogućnosti za glazbeno izražavanje kroz računalnu obradu.

## Upotreba funckijskog programiranja:

**tester.lhs:**

U ovom Haskell dokumentu prvobitno demonstriramo funkcionalnosti Euterpea paketa, programirajući jednostavne melodije tako da nadodajemo jednu notu za drugom, te ju sviramo s pomoću „play“ komande u terminalu. Nadalje testiramo mijenjanje tempa i visinu melodije, koju ćemo primjenjivati u budućim programčićima.

**randomTonal.lhs:**

Zadajemo 2 liste integera *pitches1* i *pitches2*, kojima je vrijednost jednaka visini tona, gdje *pitches1* reprezentira melodiju, a *pitches2* bass. Pri prizivanju *main* funkcije zove komandu play duet, koji u isto vrijeme svira melodiju i bass.

**lilyPondExample.hs & randomLilyPond.lhs:**

U lilyPondExample.hs dokumentu prikazujemo spremanje nota u notni zapis kao PDF file. Proširujemo tu ideju u randomLilyPond.lhs, gdje koristimo modificirani algoritam iz randomTonal.lhs kako bismo stvorili n broj nota te ih spremili u zapis u obliku koji je čitljiv LilyPond librariju, te isti spremili u PDF. Problem nailazimo u tome što taj Euterpea čita visinu note, npr. „82“, dok LilyPond čita Znakove poput „A“. Zbog toga smo napravili dvije helper funkcije koje pretvaraju output u čitljiv format za spremanje u PDF.

**BossaNovaRandom.hs:**

U ovom primjeru cilj nam je bio stvoriti improvizaciju na podlogu bossa-nova stila. Također smo modificirali nasumičnu generaciju nota tako da smo limitirali skokove između nota na dva koraka.Glavna melodija i popratna podloga sviraju u isto vrijeme.

**MidiReader.hs:**

Jednostavni program koji čita i svira MIDI file.

**MarkovChain.hs:**

Ovaj dokument koristi Markov Chain model kako bi postigao kontroliranu nasumičnost.
U ovom primjeru koristimo Emin skalu, gdje je svaka početna vrijednost nota u toj skali, a ostale vrijednosti su note u koju može prijeći i vjerojatnost da pređe u tu vrijednost.
Possible durations označuje moguće dužine nota za svaku pojedinu notu. Također smo kod pozivanja main funkcije dodali mogućnost dizanja ili spuštanja oktave za melodiju i unos tempa.

**CelullarAutomata.hs:**

U ovom dokumentu koristili smo koncept celullar automate u svrhu postizanja kontrolirane dinamike u generaciji glazbe. Dodali smo 5 popularnih algoritama (rules) od kojih svaki daje drugačiji output. Također ispisujemo vizualizaciju svakog pravila. Euterpea svira Emin skalu samo na notama gdje je polje True ili # čime se postiže kontrolirana nasumičnost.

## Prednosti i neodstatci:

Prednosti ovakvog pristupa stvaranju glazbe je velika ušteda vremena. Kvalitetnim korištenjem funkcijskog programiranja moglo bi se automatizirati dijelovi u produkciji kao što je npr. pojačanje svih kanala u mikseti, kopiranje efekata na odabrane kanale, kreiranje preseta za mix/master često korišteni instrumenata.

Mane ovakvog pristupa su te da stvaranje glazbe zahtijeva ljudski kontakt („osjećaj za glazbu“). Samim time prevelika nasumičnost nota neće uvijek stvoriti koherentni krajnji produkt tj. smislenu melodiju,ritam,pjesmu.

## Usporedba sa tradicionalnim alatima:

U usporedbi s tradicionalnim alatima za kreiranje te produciranje muzike ovakav način malo je nekonvencionalan. Naime skladanje same muzike je automatizirano čime se gubi srž muzike a to je ljudski kontakt. Naprotiv, samim time što ovakav pristup skladanju se zasnova na automatizaciji mogli bismo reći da bi unaprijeđene postojećih alata funkcijski programirani alatima uvelike umanjio vrijeme potrebno za doradu kao što mix/master. Pažljivim programiranje prateći zakone glazbe mogao bi se stvoriti kvalitetan softver koji bi u pravoj primjeni uvelike umanjio vrijeme produciranja te skladanja glazbe.

## Zaključak:

Iako funkcijsko programiranje glazbe je impresivno te uz dobru konfiguraciju može stvoriti kvalitetne rezultate možemo sa sigurnošću reći kako kvalitetan algoritam će teško zamijeniti vrhunskog skladatelja s mnogo godina iskustva.