

Univerzitet u Novom Sadu
Prirodno - matematički fakultet
Departman za matematiku i informatiku

Genetički algoritmi i muzika

-master rad-

Novi Sad, juni 2009.

Matić Dragan

Sadržaj

Predgovor	5
Algoritmi u muzici	7
Definicija algoritma	7
Upotreba algoritama u muzici.....	7
Istorijski pregled formalizovanja muzike i algoritamskog komponovanja	8
Algoritmi u staroj Grčkoj	8
Pitagorejska skala.....	9
Jednako temperirana skala – put ka formalizaciji muzike	10
Moteti.....	11
Kanonska muzika.....	11
„Muzički točak“	11
Igra pomoću muzičke kocke.....	12
Fux-ov algoritam	13
Sonate i algoritmi	13
Algoritmi u muzici u 20. vijeku	14
Muzička terminologija.....	15
Algoritamsko komponovanje muzike.....	18
Načini algoritamskog komponovanja muzike	20
Matematički modeli	20
Sistemi zasnovani na znanju.....	21
Gramatike.....	21
Sistemi koji uče.....	21
Hibridni sistemi	22
Genetički algoritmi u muzici.....	23
Genetički algoritmi	23
Osnovni dijelovi implementacije genetičkih algoritama, sa aspektima upotrebe u muzici.....	24
Kodiranje	24
Kodiranje pojedinačnih tonova	25

Kodiranje nizova tonova.....	25
Primjer predstavljanja kompozicije pomoću relativnog predstavljanja tonova	26
Fitness funkcija	28
Automatsko određivanje fitnessa.....	29
„Mjerljivi“ elementi kompozicije.....	29
Kriterijumi za fitness funkciju	31
Interaktivno određivanje fitnessa	32
Genetički operatori	32
Inicijalizacija	32
Selekcija.....	33
Ukrštanje	34
Mutacija	35
Zamjena.....	37
Genetički algoritmi i dizajniranje zvuka	38
Primjer optimizacije procesa kreiranja zvuka	39
Opis jednog algoritma za optimizovanje procesa kreiranja zvuka.....	41
Genetički algoritmi u komponovanju muzike – sopstveni primjer	42
Cilj i ideja algoritma.....	42
Softverski paket <i>JFugue</i>	43
Neki primjeri upotrebe JFugue elemenata koji su korišteni u radu.....	43
Klasa Player	43
Klasa Pattern	44
Detaljan opis algoritma – opšti principi funkcionisanja	44
Kreiranje jedinke	45
Određivanje fitnessa jedinke	46
Formiranje i organizacija populacije	49
Genetski operatori korišteni u algoritmu.....	49
Mutacije	49
Selekcija.....	50
Konkretno rješenje.....	50
Implementacija.....	51
Klasa Jedinka – osnovni elementi.....	51

Klasa Populacija – osnovni elementi	52
Glavni program.....	53
Primjeri generisanih jedinki	53
Primjeri generisanih „main stream“ kompozicija	53
Specijalni primjerci jedinki	55
Zaključak.....	59
Literatura.....	61

Predgovor

Genetički algoritmi su dio veće grupe algoritama u vještačkoj inteligenciji (eng. *artificial intelligence*). Obezbeđuju moćnu tehniku pretraživanja velikog, često neuređenog prostora dozvoljenih rješenja. U komponovanju, primjena genetičkih algoritama se svodi na određivanje odgovarajućeg skupa dozvoljenih rješenja (kompozicija ili njenih dijelova) i pretraživanje tog skupa, u cilju pronalazjenja dovoljno dobrog rješenja [1]. U procesu kreiranja zvuka, genetički algoritmi predstavljaju alat koji omogućava dobijanje i analizu različitih zvukova preko kombinovanja različitih parametara koji utiču na period oscilovanja, povećanje/smanjenje amplitude i slično.

U svom poznatom i često citiranom članku [2], Biles ovaj proces upravo opisuje kao **istraživanje** muzičkog prostora koji sadrži sve potencijalne varijante. Muzički zadaci kao što su komponovanje, aranžiranje ili improvizacija uvode i u značajnoj mjeri koriste pretraživanje. Kompozitori traže odgovarajuće akorde koji se uklapaju u melodiju, ili pravu melodiju koja bi pratila niz akorada. Aranžeri traže odgovarajuće glasove i kontrapunktualnosti za sastavne dijelove kompozicije, a improvizatori pokušavaju pronaći odgovarajuće fraze koje se uklapaju, na primjer, u posmatrani skup promjena akorada. Ovdje već dolazimo do kritične tačke i ključnog problema u algoritamskom komponovanju, a to je procjena kvaliteta primjerka dobijenog pretraživanjem, jer u realnom okruženju presudnu ulogu igra estetski osjećaj kompozitora.

Ovaj problem nije lako prevazići. Stoga za cilj algoritamskog komponovanja i ne treba postavljati pronalazjenje optimalnog rješenja (pitanje je da li ono i postoji), već se treba bazirati na **eksploataciju** prostora svih kompozicija, u cilju prepoznavanja potencijalno dobrih rješenja, i mogućnosti odabira jednog ili više njih.

Uzimanje postojećih muzičkih ideja, šablona ili sistema i odgovarajuće promjene i varijacije koje se vrše na njima često predstavljaju polaznu osnovu u komponovanju. Muzičari u različitim stilovima i žanrovima koriste ovakav sistem komponovanja: oblikovanje novih ideja na osnovu postojeće početne tačke može se posmatrati kao uobičajen proces i zajednička karakteristika mnogih muzičkih stilova.

Genetički algoritmi odgovaraju ljudskom procesu istraživanja. Prema [3], dva ključna procesa u genetičkim algoritmima, mutacija i ukrštanje zajedno sa selekcijom formiraju osnovu za inovaciju koja se veoma slično ponaša kao i način ljudskog inovativnog procesa:

Selekcija + Mutacija = Neprekidno poboljšanje

Selekcija + Ukrštanje = Inovacija.

Ovi objektivni zaključci o ulozi i namjeni genetičkih algoritama umnogome objašnjavaju opravdanost upotrebe genetičkih algoritama u komponovanju, ali i bilo kom drugom procesu koji je povezan sa umjetnošću i drugim kreativnim radom, i za koji su karakteristični neprekidno (kontinuirano) poboljšanje rezultata i inovacija. Ovo, međutim, ne znači da upotreba genetičkih algoritama (ili nekog drugog metoda) u komponovanju treba da konkuriše kreativnim sposobnostima čovjeka. Jasno je da se nijednim algoritmom ne može komponovati klavirski koncert, niti se mogu dobiti predivne teme najčuvenijih

kompozitora. Ono što se ovakvim sistemima može dobiti treba da ostane u domenu uže primjene: za eksperimentalu muziku, semplovanje, aranžiranje, kao tehnička muzika kao podloga za animacije, simulacije i slično.

U prvom poglavlju ovog rada su kroz istorijski pregled objašnjeni osnovni motivi i načini upotrebe algoritama u muzici kroz dugu istoriju izučavanja muzike i matematičkih pravilnosti koje u njoj vladaju.

U drugom poglavlju su navedeni i ukratko objašnjeni najvažniji muzički termini koje autor koristi u nastavku rada.

U trećem poglavlju su navedeni i ukratko opisani načini algoritamskog komponovanja.

Četvrto poglavlje je posvećeno objašnjenju funkcionisanja genetičkih algoritama uopšte, kao i načinima upotrebe genetičkih algoritama u komponovanju muzike.

U petom poglavlju je izneseno razmatranje koje je nezavisno od ostalih poglavlja, a tiče se primjene genetičkih algoritama u dizajniranju zvuka.

U šestom poglavlju autor prezentuje sopstveni primjer genetičkog algoritma koji se koristi za komponovanje kratkih kompozicija dužine četiri četvoročetvrtinska takta. Izložen je opšti princip funkcionisanja algoritma, metodologija razvoja, uopštena analiza rješenja, kao i analiza dobijenih konkretnih rješenja, u odnosu na zadate parametre. Neka rješenja su prezentovana u notnom zapisu.

Posljednja dva poglavlja ovog rada su zaključak i spisak korištenih referenci.

Algoritmi u muzici

Iako se u današnje vrijeme upotreba algoritama u većini slučajeva vezuje za primjenu računara, algoritamsko rješavanje problema (zadataka) ne zahtijeva i njihovu obaveznu upotrebu. U širokoj i raznovrsnoj literaturi koja se bavi problematikom algoritama, često se navode primjeri koji nisu ni u kakvoj vezi sa računarima. Tako, algoritam kao skup pravila koji dovodi do rješenja može biti i refleksna radnja (kao što je treptanje oka) ili radnja prikazana i razmotrena pomoću nekog drugog, fizički opipljivog sistema i alata (čuveno "ubacivanje slona u frižider", ili proces kuvanja čaja).

Stoga se i upotreba algoritama u muzici ne mora usko vezivati samo za kompjuterske algoritme, te ima smisla napraviti istorijski pregled korištenja algoritama u muzici, posebno algoritamskog komponovanja.

Definicija algoritma

Neformalno, algoritam je bilo koji dobro definisan skup procedura, koje uzimaju neku vrijednost ili niz vrijednosti kao skup ulaznih podataka¹, a kao rezultat računanja daje neki niz izlaznih podataka. Algoritam tako predstavlja niz definisanih koraka u računanju, koji niz ulaznih transformišu u niz izlaznih podataka.

Algoritam takođe možemo posmatrati kao dobro opisan problem. Definicija samog problema daje vezu između ulaznih i izlaznih podataka. Algoritam opisuje specifične procedure u računanju pomoću kojih se postiže data veza [4].

Izvorno, riječ algoritam potiče od arapskog matematičara Al Hvarizmi-ja² [5] kome se, između ostalog pripisuje i uvođenje arapskih brojeva. Konstrukcija riječi potiče zapravo od pogrešne interpretacije latinskog izraza "*algorismi dixit*" (alhvarizmi govori)³. Riječ "*algorismi*" podsjeća na množinu od "*algorismus*", koja je opet slična grčkoj riječi *arithmos* (ἀριθμός) što znači broj. Otuda se za termin algoritam vezuje i značenje "metod za računanje" [6].

Upotreba algoritama u muzici

Algoritmi u muzici se koriste na onim mjestima gdje primjena niza pravila ili instrukcija može dovesti do (kvalitetnog) rješenja. Algoritme u muzici srećemo prilikom sinteze zvuka, semplovanja⁴, prepoznavanja muzičkih djela ili komponovanja. Dok se kod prve tri aktivnosti algoritmi prirodno nameću kao načini rješavanja problema (pretraživanje po stablima, nizovima ili neuređenim strukturama, te primjena strogih pravila koji opisuju pojedine korake algoritma), u komponovanju muzike algoritmi pokušavaju zamijeniti ono što se zapravo smatra isključivo ljudskom djelatnošću: komponovanje kao umjetnička aktivnost podrazumijeva slobodan izbor tonova pomoću kojih kompozitor iskazuje osjećanja, raspoloženja, namjere ili inspiracije. Zagovornici algoritamske muzike upravo taj slobodan izbor smatraju propisanim pravilima koja se relativno lako mogu interpretirati odgovarajućim nizom instrukcija. Većina

¹ Ovaj skup teoretski može biti i prazan.

² Abu Ja'far Muhamad ibn Musa-al-Hvarizmi (oko 780-850), persijski matematičar, astronom i geograf

³ Prema [6] riječ algoritam se formira od grčke riječi "*arithmos*" - broj i arapske riječi *algorism* što znači niz brojeva

⁴ eng. sampling – uzimanje jednog dijela nekog muzičkog djela zbog upotrebe u drugim kompozicijama

kompozitora primjenjuje određena pravila prilikom komponovanja, tj. nizove ili skupove instrukcija, te se tako bilo koje komponovanje na neki način može svesti na algoritamsko. Sa druge strane, nedostatak ljudskog faktora u (automatskom⁵) algoritamskom komponovanju dovodi do pojave velike količine objektivno loše i neupotrebljive muzike.

Stoga se dosta zagovornika algoritamskog komponovanja odlučuje za uključenje ljudskog faktora rasuđivanja u toku izvršenja algoritma. Ovakav oblik komponovanja se zove interaktivno komponovanje, gdje se u ključnom trenutku procjene kvaliteta kompozicije (ili njenog dijela) uključuje uticaj čovjeka. Pokazuje se da ovakav pristup često daje bolje rezultate od automatskog komponovanja, jer čak i veliki broj ubačenih pravila i ograničenja u algoritmima ne može dovoljno dobro procijeniti kvalitet melodije.

Istorijski pregled formalizovanja muzike i algoritamskog komponovanja

U dugoj istoriji proučavanja matematičkih pravilnosti u muzici, kompozitori i muzički teoretičari su koristili razne tipove algoritama. Ograničenja u komponovanju, određivanje pravilnosti koje vladaju u harmoniji, kao i praćenje odgovarajućeg muzičkog stila umnogome su doprijenili algoritamskom pristupu. Ako pretpostavimo da prilikom komponovanja kompozitor treba preduzeti konačan broj koraka, gdje se nailazi na jasna ograničenja (broj raspoloživih tonova, matematičke pravilnosti ritma i tempa, harmonija), jasno je da je komponovanje muzike u nekom opštem smislu u stvari poštovanje unaprijed definisanog niza pravila, što algoritam po svojoj definiciji upravo jeste. Na primjer, svođenje svih frekvencija tonova unutar jedne oktave sa 12 tonova već samo po sebi nameće pravila koja značajno smanjuju skup dozvoljenih tonova a samim tim u nekoj mjeri i kreativnost kompozitora. Kanonska muzika (na primjer fuge) ima osobinu „kontrapunktualnosti“⁶ koja povlači stroga ograničenja u komponovanju i veoma ograničavajuće algoritme.

Algoritmi u staroj Grčkoj

Matematičke pravilnosti koje vladaju u muzici kroz istoriju su predmet izučavanja i matematičara i muzičkih teoretičara. U evropskog civilizaciji izučavanje muzike kao teoretske discipline i prvi pokušaji formalizacije muzike, počinju u Staroj Grčkoj. Smatra se da ovakvi korijeni potiču još od vremena Pitagore. Iako od ovog grčkog filozofa koji je živio u 6. vijeku prije nove ere nije ostalo pisanih tragova, na osnovu kasnijih djela pretpostavlja se da se od njegovog vremena izučavaju odnosi između skladnih tonova – konsonanata [7]. Takvi odnosi su opisivani kao odnosi između (malih) brojeva, 1, 2, 3 i 4. Na primjer, zna se da za jedan osnovni ton frekvencije f imamo niz "skladnih" tonova, koji su frekvencije $2f$, $3f$, $4f$... Vidi se da su ovakvi tonovi sa osnovnim u proporciji 1 : 2; 1 : 3; 1 : 4. Takođe, tonovi koji su sa osnovnim tonovima u proporciji 2 : 3 ili 3 : 4 takođe su prijatni ljudskom uhu (kvinta i kvarta). Izučavanje odnosa između frekvencija tonova, te određivanje i ocjenjivanje "dobrih" i "loših" intervala se, naravno, i danas smatraju fundamentalnim elementima u algoritamskom komponovanju. Za Pitagoru i njegove sljedbenike svijet muzike je imao mnogo šire značenje nego danas, kada se muzika prvenstveno smatra umjetnošću. Muzika je u to doba bila neodvojiva od brojeva, koji su smatrani ključem čitavog duhovnog i

⁵ Pod automatskim komponovanjem se smatra ono koje isključuje bilo kakvo učešće ljudskog faktora u toku izvršenja algoritma.

⁶ kontrapunktualnost, lat. *punctus contra punctum* – „tačka nasuprot tački“, predstavlja vezu između dvije ili više dionica koje su međusobno zavisne

fizičkog univerzuma.[8] Tako je sistem muzičkih zvukova i ritmova, uređen odgovarajućim brojevima zapravo predstavljao personifikaciju skladnosti čitavog univerzuma.

Pitagorejska skala

Pitagorejska skala predstavlja jednu od prvih pokušaja formalizacije predstavljanja i zapisa muzičkih tonova preko odgovarajućih frekvencija i odgovarajućih intervala preko odnosa tih frekvencija. Princip računanja odnosa između tonova i davanje prioriteta određenim (skladnim) odnosima prenesen je i do današnjeg vremena, gdje se, kao što će u nastavku biti prikazano, odgovarajućim skladnim intervalima daje veća (težinska) vrijednost ili veća frekvencija pojavljivanja.

Pitagorejska skala je zasnovana na jednostavnim principima. Posmatra se osnovni ton sa frekvencijom f . Za oktavu viši ton ima frekvenciju $2f$, dok čista kvinta ima frekvenciju $\frac{3}{2}f$. Tako na primjer, ako za osnovni ton uzmemo ton C1 i dodijelimo mu frekvenciju 1, tada ton G1 ima frekvenciju $\frac{3}{2}$. Od tona G1 računamo frekvenciju tona D2, kao $\frac{3}{2} \cdot \frac{3}{2}$, dok onda ton D1 (za oktavu niži od D2) ima frekvenciju $\frac{3}{2} \cdot \frac{3}{2} \cdot \frac{1}{2} = \frac{9}{8}$. Dalje, ton A1 je frekvencije $\frac{9}{8} \cdot \frac{3}{2} = \frac{27}{16}$, ton E2 $\frac{27}{16} \cdot \frac{3}{2}$, odnosno E1 $\frac{27}{16} \cdot \frac{3}{2} \cdot \frac{1}{2} = \frac{81}{64}$, ton H1 $\frac{81}{64} \cdot \frac{3}{2} = \frac{243}{128}$, ton Fis1 $\frac{243}{128} \cdot \frac{3}{2} \cdot \frac{1}{2} = \frac{729}{512}$ i ton Cis1 $\frac{729}{512} \cdot \frac{3}{2} \cdot \frac{1}{2} = \frac{2187}{2048}$. Ako posmatramo niz tonova

$$C1 - G1 - D2 - A2 - E3 - H3 - Fis3(Ges3) - Cis4(Des4) - Gis4(As4) - Es5 - B5 - F6 - C7$$

vidimo da se nakon 12 tonova ponovo vraćamo na početak (to je tzv. kvintni krug). Može se ići i u drugom smjeru. Frekvencija tona F1 se može izračunati tako što se uzme kvinta "nadole", pa onda oktava "gore", tj. $\frac{2}{3} \cdot 2 = \frac{4}{3}$. Ako nastavimo postupak unazad, od tona F1 dobijamo ton B1 (sniženo H1) $\frac{4}{3} \cdot \frac{2}{3} \cdot 2 = \frac{16}{9}$. Dalje, ton Es1 $\frac{16}{9} \cdot \frac{2}{3} = \frac{32}{27}$, ton As1 $\frac{32}{27} \cdot \frac{2}{3} \cdot 2 = \frac{128}{81}$ i konačno, ton Des1 (odnosno Cis1) $\frac{128}{81} \cdot \frac{2}{3} = \frac{256}{243}$.

Može se primijetiti da jednim i drugim sistemom ne dobijamo potpuno isti odnos za tonove Des i Cis, (iako to zapravo treba da budu isti tonovi), a to je upravo posljedica nesavršenosti Pitagorejske skale. U odnosu na početni ton, Pitagorejska skala daje savršenu skladnost, te, ako bi se samo to gledalo, ne bi bilo potrebe za uvođenjem bilo kakvog drugog skaliranja. Međutim, problem nastaje ukoliko na instrumentu koji je naštimovan po ovom principu izaberemo neki drugi ton kao osnovni. U muzici, ovakva pojava se zove modulacija, ili transpozicija, u zavisnosti da li je riječ o pomjeranju tonaliteta unutar jedne, ili je riječ o dvije različite kompozicije. Ovakve situacije su se, prije uvođenja jednako temperirane skale rješavale na vrlo nepraktičan način: između dvije kompozicije se vršilo prenaštimanje instrumenata! [9]

Ova konfuzija ima jednostavno matematičko objašnjenje. Ako se krene recimo od tona C1 i napravi niz čistih kvinti, dobija se ton C7 koji je za 7 oktava viši od polaznog, tj taj ton ima frekvenciju $2^7 = 128$. Sa druge strane, taj ton mora imati frekvenciju koja se računa stepenovanjem kvinte 12 puta, tj. $\left(\frac{3}{2}\right)^{12} = 129.7463$. Očigledno, ova dva broja nisu jednaka.

Jednako temperirana skala – put ka formalizaciji muzike

Jednako temperirane skale su skale kod kojih je odnos između bilo koja dva uzastopna tona konstantan. U evropskoj kulturi, uobičajeno je da se koristi dvanaest-tonska jednako temperirana skala. Matematički gledano, nije teško odrediti tačan odnos između dva tona u bilo kojoj jednako temperiranoj skali.

Ograničimo se na dvanaest-tonsku jednako temperiranu skalnu. Pretpostavimo da je osnovni ton frekvencije a , a b je koeficijent rasta u skali. Ukoliko želimo jednake odnose između uzastopnih tonova, treba da bude svejedno da li ćemo do prvog tona susjednog osnovnom u jednoj oktavi iznad doći tako što ćemo duplirati njegovu frekvenciju iz osnovne oktave, ili se 12 puta (odnosno 13 puta od osnovnog tona) pomjeriti pomoću koeficijenta rasta.

Treba da važi:

$$2(ab) = ab^{13}$$

Odnosno,

$$b = \sqrt[12]{2}$$

Očigledno je da se, bez obzira kolika se preciznost uzme, odnosom dva prirodna broja (racionalnim) brojem postiže samo manje ili više dobra aproksimacija jednog iracionalnog broja. Razlike u frekvencijama koje se dobijaju između odnosa tonova u jednako temperiranoj skali i drugim skalama, zasnovanih na racionalnim brojevima kao odnosima između tonova, mogu se predstaviti na dva načina: kao standardna relativna razlika ili kao devijacija koja se mjeri u centima. Cent se definiše kao 1200 puta logaritam sa osnovom 2 od posmatranog odnosa frekvencija. Tako je u jednako temperiranoj skali oktava, koja se predstavlja sa ukupno 1200 centi, podijeljena na 12 polutona, gdje svakom polutonu odgovara 100 centi. Devijacija d_i , za i ti ton u skali, koji je sa početnim u odnosu k_i je:

$$d_i = i \cdot 100 - 1200 \log_2 k_i.$$

Devijacija od na primjer, 5 centi ili jedne dvadesetine polutona odgovara oko 0.3% standardnog relativnog odstupanja, što je, za osobu sa istančanim sluhom gornja granica tolerancije. U tabeli 1 prikazane su devijacije (odstupanja) tonova iz Pitagorejske skale zasnovane na cjelobrojnim proporcijama u odnosu na tonove jednako temperirane skale. Kolona označena sa "JTS" (jednako temperirana skala) sadrži odnose frekvencija u jednako temperiranoj skali. Kolona "Odnos" predstavlja racionalni odnos frekvencija Pitagorejske skale, dok kolona "Dec. vr." predstavlja decimalnu vrijednost tog odnosa. U posljednjoj koloni ("Dev") navedena je devijacija date racionalne aproksimacije u odnosu na jednaki temperament.

Tabela 1: Ocjena greške tonova u Pitagorejske skale u odnosu na jednako temperiranu.

Ton	Interval	JTS	Odnos	Dec. vr.	Dev (centi)
0	Prima	1.000	1:1	1.000	0.000
1	Mala sekunda	1.059	2187:2048	1.068	-13.685
2	Velika sekunda	1.122	9:8	1.125	-3.910
3	Mala terca	1.189	32:27	1.185	5.865
4	Velika terca	1.260	81:64	1.266	-7.820

Ton	Interval	JTS	Odnos	Dec. vr.	Dev (centi)
5	Čista kvarta	1.335	4:3	1.333	1.955
6	Prekomjerna kvarta	1.414	729:512	1.424	-11.730
7	Čista kvinta	1.498	3:2	1.500	-1.955
8	Mala seksta	1.587	128:81	1.580	7.820
9	Velika seksta	1.682	27:16	1.688	-5.865
10	Mala septima	1.782	16:9	1.778	3.910
11	Velika septima	1.888	243:128	1.898	-9.775
12	Oktava	2.000	2:1	2.000	0.000

Formalizacija tonova i intervala se kod starih Grka ipak više zadržavala u teoriji, dok je striktna primjena na izvođenje muzičkih djela najvjerojatnije upitna, jer je grčka muzika većinom bila zasnovana na improvizacijama [8]. Tako, iako su propisana pravila u muzici koja su uticala na proces komponovanja, muzičari nisu ni na koji način izbačeni iz procesa odlučivanja. Starogrčka muzika nije alogritamski komponovana, ali je, sa druge strane, istorijski važna u kasnijim pokušajima ili tendencijama formalizovanja muzike.

Moteti

U periodu od 1300 do 1450 godine, sa početkom Ars Nova⁷ pokreta u Francuskoj kompozitori su koristili izoritmičku tehniku za komponovanje muzike, gdje se ritam u kompoziciji periodično ponavljao prema unaprijed definisanom sistemu. [10]

U nekim motetima vidljiva je i upotreba poznatog odnosa zlatnog presjeka, gdje su dužine odgovarajućih dijelova u tom odnosu. Ipak, u novijim radovima, i pored nekih pokušaja [11] ovaj odnos nije prepoznat kao značajan ili pouzdan, u smislu postizanja bolje harmonije i uklapanja dijelova kompozicije, kako u prošlosti, tako i u modernim sistemima algoritamskog komponovanja.

Kanonska muzika

Pojava kanonske muzike krajem 15. vijeka je klasičan primjer algoritamskog komponovanja. Najčešće korišten metod je bio pisanje melodije za jedan glas, dok su se ostali glasovi izvodili iz njega. Uputstvo ili pravilo na osnovu kog su izvođeni ostali glasovi je nazvan *kanon*, što i znači pravilo ili zakon. Na primjer, drugi glas je mogao kasniti za prvim nekoliko taktova, ili je mogao biti niz inverznih tonova u odnosu na osnovni. Ovakva pravila imitacije daju algoritamski pristup, a sam kompozitor je sklonjen iz dijela procesa komponovanja.

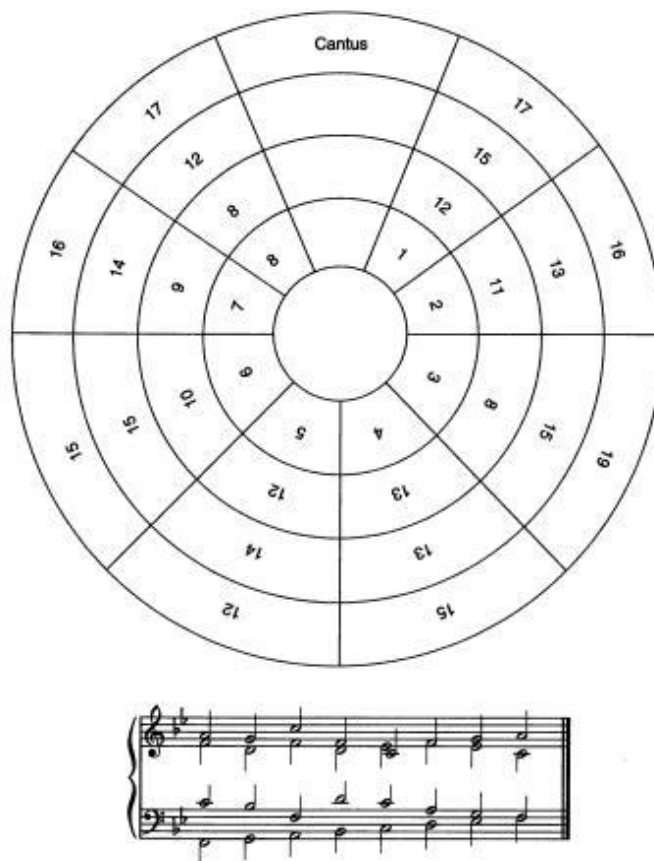
„Muzički točak“

1660 godine italijanski kompozitor Bontempi⁸ je zamislio točak (eng. *rota*), pomoću kog se može komponovati bez ikakve muzičke – umjetničke dimenzije. Svaki nivo, od unutra ka spolja sadrži brojeve

⁷ Ars Nova (lat.) – Nova umjetnost, muzički pokret u srednjovjekovnoj Francuskoj

⁸ Giovanni Andrea Bontempi (1624-1705), italijanski kompozitor je “muzički točak” opisao u svom djelu Novi metod komponovanja četiri glasa

koji predstavljaju stepene tonova u miksolidijskoj⁹ skali za jedan od četiri glasa nove kompozicije. Na Slici 1 prikazani su stepeni tonova u miksolidijskoj skali i odgovarajući notni zapis kompozicije.



Slika 1: Muzički točak

Skala je zasnovana na tonu F, odnosno B duru. Iako ovaj način komponovanja ne daje velike mogućnosti, sa istorijskog aspekta značajan je kao jedna od preteča algoritamskom pristupu komponovanju. Kod muzičkog točka imamo predstavljanje tonova odgovarajućim brojevima (indeksima), počev od jednog, referentnog tona, koji je indeksiran jedinicom [6]. Ovakav način predstavljanja tonova viđaće se i kasnije.

Igra pomoću muzičke kocke¹⁰

Ideja muzičke kocke je kombinacija unaprijed zapisanih (komponovanih) dijelova na kontrolisan slučajan način, u cilju komponovanja muzičkog komada. Slučajan odabir dijelova zasnovan je na bacanju kockice. Standardno se bacaju dvije kockice ukoliko se "komponuje" menuet, odnosno jedna ukoliko se "komponuje" "trio". U slučaju bacanja dvije kockice uzima se zbir dobijenih rezultata bacanja. Na osnovu dobijenog rezultata iz definisane tablice se bira odgovarajući dio koji se ubacuje u melodiju.

⁹ eng. *mixolydian*, način organizacije nota kao u durskoj ljestvici, s tim što se za polazni ton uzima onaj koji je peti po redu u odgovarajućoj ljestvici. Na primjer, ako uzmemo F dur kao ljestvicu, tada miksolidijska skala glasi CDEFGABC.

¹⁰ Originalno, na njemačkom jeziku, Musikalisches Würfelspiel

Najpoznatija muzička kocka potiče od W.A. Mozarta. Prvi put je objavljena dvije godine nakon njegove smrti (1793) i iako nije pronađena direktna referenca koja upućuje da je riječ o Mozartovom djelu, prema Kehelu¹¹, nesumnjivo je da je autor pomenute muzičke kocke upravo Mozart.

Postoji ukupno 176 mjera¹² za menuet (11 mogućnosti za svaku od 16 mjera), odnosno 96 mogućnosti za trio (6 mogućnosti za svaku od 16 mjera). Na osnovu tabele u kojoj je naveden raspored korištenja kompozicija u odnosu na rezultat i redni broj bacanja, određuje se koja mjera se u kom koraku treba uzeti. Lijepa ilustracija upotrebe Mozartove kocke može se pronaći na sajtu¹³ autora [12].

Prostim kombinovanjem gore navedenih mogućnosti, dobijamo da je ukupno moguće komponovati $11^{16} \cdot 6^{16} = 129629238163050258624287932416$ različitih kompozicija koje se sastoje od jednog menueta i jednog trija.

Fux-ov algoritam

Jedan od tipičnih primjera algoritamskog pristupa komponovanju dao je Fuks¹⁴, u svom djelu, 1725. godine. Fuks je, opisujući mnoge osnovne kontrapunktualne tehnike tonalne muzike i propisujući pravila u komponovanju, uticao na mnoge kasnije kompozitore da se pozabave ovakvim pristupom komponovanju, među kojima je, navodno, bio i Johann Sebastian Bach. Fuks je ustanovio sljedeća pravila [6]:

1. Od jednog savršenog konsonanta¹⁵ do drugog savršenog konsonanta mora se doći u suprotnom ili nedirektnom (eng. *oblique*) kretanju¹⁶,
2. Od savršenog konsonanta do nesavršenog konsonanta¹⁷ može se doći pomoću bilo kog kretanja,
3. Od nesavršenog konsonanta do savršenog konsonanta mora se doći u obrnutom ili nedirektnom kretanju,
4. Od nesavršenog konsonanta do nesavršenog konsonanta može se doći pomoću bilo kog kretanja.

Vrijedi spomenuti i činjenicu da je Kehel izvršio katalogizaciju Fuksovih dijela (već je pomenuta Kehelova katalogizacija Mozartovog opusa), čime je, sredinom devetnaestog vijeka, Fuksova muzika ponovo oživjela.

Sonate i algoritmi

U organizaciji sonata postoje pravilnosti koje se prije svega odnose na podjelu same kompozicije na više dijelova. Po pravilu, većina sonata se sastoji od nekih od sljedećih dijelova: uvoda, izlaganja osnovne teme (ekpozicija), razrade, rekapitulacije i kode. Organizovanje sonate po ovom unaprijed propisanom

¹¹ Ludwig Alois Ferdinand Ritter von Köchel, njemački muzikolog, koji je 1862. godine objavio sistematizovan Mozartov opus

¹² Mjeru ovdje možemo shvatiti kao sempl – kratku melodiju koja služi za kreiranje većih kompozicija

¹³ <http://sunsite.univie.ac.at/Mozart/dice/>

¹⁴ Johann Joseph Fux (1660–1741), austrijski (štajerski) kompozitor i muzikolog

¹⁵ Savršeni konsonanti su čisti intervali: prima, kvarta, kvinta i oktava

¹⁶ Postoje tri vrste kretanja: direktno, kada obje dionice idu u istom smjeru, obrnuto, kada dionice idu u suprotnim smjerovima i *nedirektno*, kada jedna dionica (melodija) ide u nekom smjeru, a druga stoji

¹⁷ Nesavršeni konsonanti su mala i velika terca te mala i velika seksta

sistemu bi moglo da da osnovnu pretpostavku za algoritamski pristup. Ipak, sonata je zasnovana na “slobodi komponovanja”, što i daje njenu pravu ljepotu. Navedena ograničenja u samoj organizaciji, kao i dodatna ograničenja (upotreba kvintnog kruga kao osnove za promjenu tonaliteta, odnosi između dužina dijelova i slično), ipak ne mogu dovoljno dobro usmjeriti i determinisati potencijalni algoritam. S obzirom na postignute rezultate raznih autora u algoritamskom komponovanju, sonate kao kompleksna djela najvećih kompozitora ostaju van domena uspješnih rješenja.

Algoritmi u muzici u 20. vijeku

Upotrebom računara za algoritamsko komponovanje, u dvadesetom vijeku se broj primjera upotrebe algoritama u muzici drastično uvećava. Bilo je mnogo eksperimentalnih (manje ili više uspješnih) pokušaja primjena raznih algoritamskih pristupa. Jedan od najpoznatijih avangardista dvadesetog vijeka Džon Kejdž¹⁸ je pravio kompozicije koristeći razne načine slučajnog generisanja tonova. Na primjer, muzika je formirana na osnovu šahovske partije, gdje je svaki potez uzrokovao odgovarajući zvuk (partija je igrana na foto – osjetljivoj šahovskoj tabli); ili, raspored nota na notnom sistemu je zasnovan na rasporedu zvijezda na nebu.[13]

Upotrebom standardnih dvanaest – tonskih ljestvica prostor svih mogućih tonova je “ukroćen” u konačne okvire. Većina pokušaja se tako svodi na određivanje i kontrolu svih (ili dovoljnog broja) parametara koji utiču na kvalitet kompozicije i pronalaska dovoljno dobrog rezultata. Kao predkompozitorski proces, iz čitavog skupa se prvo izdvajaju “dozvoljene” kompozicije (na primjer one koje imaju unaprijed propisan opseg tonova, ritmova i slično), te se među njima, nekom kompjuterskom metodom koja oponaša proces komponovanja od strane čovjeka pokušava pronaći zadovoljavajuće rješenje.

¹⁸ John Milton Cage Jr. (1912 –1992), američki kompozitor

Muzička terminologija

U ovom dijelu daćemo definicije osnovnih termina iz muzike, zasnovane na [14].

Visina tona ili *ton* je osnovni pojam u muzici. Visina tona se može posmatrati kao subjektivan osjećaj koji čuje ljudsko uho, ali i kao objektivna vrijednost (na primjer, vrijednost frekvencije odgovarajućeg zvučnog talasa). Postoji relativno i apsolutno određivanje visine tona. Relativno određivanje visine tona je zasnovano na određivanju visine u odnosu na neki početni ton (na primjer, ton D4 je viši od tona C4). Apsolutna visina predstavlja objektivnu i konstantu vrijednost visine tona (na primjer frekvencija (koja jednoznačno indukuje i visinu) tona A4 je 440Hz).

Tonovi se zapisuju *notama*, koje u evropskom standardnom sistemu predstavljaju 12 podjednako po visini udaljenih *polustepena*. *Polustepen* je najmanji praktično upotrebljivi razmak između dva tona. U jednako temperiranoj dvanaest-tonskoj skali propisan je standardni odnos frekvencije susjednih polutonova i iznosi $^{12}\sqrt{2}$. *Tonski sistem* je tako sačinjen od geometrijskog niza frekvencija tonova sa prvim članom jednakim jedan i količnikom $^{12}\sqrt{2}$. Skala sadrži sedam osnovnih nota („c“, „d“, „e“, „f“, „g“, „a“, „h“). Nakon note „h“ dolazi ponovo nota „c“ koja se od početne note „c“ razlikuje po visini, odnosno ima dvostruko veću učestanost od početne, pa ona ujedno i započinje novi tonski niz. *Oktava* sadrži dvanaest polutonova, sedam osnovnih i pet izvedenih, analogno crnim i bijelim tipkama na klaviru. Da bi se tonski nizovi međusobno razlikovali, notama iz odgovarajućeg niza dodaju se broježani indeksi, npr „c1“, „c2“, „c3“ itd. Oktava je, prema tome, niz od osam tonova (npr. „c-c1“), sastavljena od dvanaest polutonova.

U formiranju muzičke kompozicije neophodan element je *trajanje*, bilo samog tona, bilo *pauze* između tonova. Sistem označavanja trajanja koji se danas najčešće koristi je takav da je svako naredno kraće trajanje bilo tona, bilo pauze, upola manje od prethodnog. Tako se sistem trajanja nota sastoji od geometrijskog niza broja dva sa količnikom dva (*cijela nota*, *polovina note*, *četvrtina*, *osmina*, *šesnaestina note* itd.). Trajanje cijele note je četiri *otkucaja*, sljedeće manje trajanje iznosi dvije (za polovinu), jednu dobu (za četvrtinu) itd. *Ritam* je povezan sa vremenskim trajanjem, kako trajanjem tonova i pauza, tako i rasporedom njihovog nastupanja. Vrijeme trajanja tonova i vrijeme nastupanja tonova u melodiji definiše ritam čija je osnovna mjerna jedinica *takt*. Najčešće se muzička djela tako organizuju da imaju svoj ritam i takt, ali postoje djela kod kojih takt nije konstantan, kao i djela koja nemaju svoj takt.

Boja zvuka je mjera kojom se određuje jedinstveni identitet (raspoznatljivost) pojedinog izvora zvuka. Fizički je boja zvuka povezana sa sadržajem viših harmonika zvuka uz osnovni ton.

Jačina zvuka je mjera snage koju zvuk posjeduje, a fizički odgovara veličini amplitude zvuka.

Notni zapis je način pisanja nota. Sastoji se od *notnog sistema*, koji ima pet vodoravnih linija, *ključa* (violinskog ili bas), kojim se referiše raspored nota po linijama, te mjere, koja se nalazi na samom početku notnog zapisa u obliku dva broja postavljena jedan iznad drugog. Mjera određuje koliko se doba nalazi u jednom taktu. Na primjer, kada se koristi mjera četiri-četvrtine, tada jedan takt traje četiri dobe, odnosno isto koliko i cijela nota.

Svaka osnovna nota može biti povišena, pri čemu joj se dodaje nastavak „-is“, odnosno snižena, kada joj se dodaje nastavak „-es“ (npr. „cis“, „dis“, „eis“ itd). Povišavanjem note (stepenovanjem) njen *stupanj* se povećava za pola stepena, odnosno snižavanjem smanjuje za pola stepena. U standardnom (klavirskom) zapisu, povišena nota „e“, tj. „eis“ jednaka je noti „f“, kao i što je povišena nota „h“, tj. „his“ jednaka noti „c1“. Analogno, snižene note „f“ i „c“ su „e“ i „h“. Pri pisanju notnog zapisa povišenog tona koristi se znak *povisilica* (#), za zapis sniženog tona znak *snizilica* (b), a kada se stepenovanje ukida, koristi se znak *razrejšilica* (#̣).

Organizovanjem tonova u horizontalnoj sekvenci, tako da nailaze jedan poslije drugog, nastaje *melodija*. Vertikalnim organizovanjem tonova, kada se više tonova čuje istovremeno dobijamo *akord*, dok simultanom upotrebom (sviranjem) niza akorada koji daju subjektivno prijatan zvuk dobijamo *harmoniju*.

Kada se tonovi u oktavi nanižu ulazno ili silazno bez ponavljanja već uvrštenih tonova, dobija se *ljestvica*. Dvije najčešće korištene ljestvice su durska i molska. U durskoj ljestvici polutonovi su između 3. i 4. i 7. i 8. stepena, dok su u molskoj ljestvici polustepeni na drugom i petom stepenu (prirodna molska skala), na drugom, petom i sedmom (harmonijska), i na drugom, šestom i sedmom (melodijska). *Tonalitet* je sistem organizacije tonova u kome su odnosi između tonova zasnovani na odnosima sa izabranim početnim tonom (*tonikom*). *Tonika* je osnovni ton tonaliteta, po kome se naziva i ljestvica tog tonaliteta i on je ujedno središte tonaliteta.

Razmak između visine dvije note, bilo da njihovi tonovi zvuče istovremeno, ili jedan poslije drugog, naziva se *interval*. Klasifikuju se u *konsonantne* intervale, koji stvaraju subjektivan osjećaj prijatan za ljudsko uho, i *disonantne* intervale, koji stvaraju subjektivan osjećaj napetosti prilikom slušanja. U standardnom evropskom sistemu definiše se osam intervala, između osam (plus jedna) osnovnih nota u oktavi: *prima*, *sekunda*, *terca*, *kvarta*, *kvinta*, *seksta*, *septima* i *oktava*. Prima interval je trivijalan, jer se odnosi na isti ton. Intervali se dalje klasifikuju na:

- *čiste*, koji se javljaju u samo jednoj veličini razmaka između tonova (eventualno sa jednim izuzetkom),
- *velike* i *male*, koji se približno podjednako učestalo javljaju u dvije veličine različite za pola stepena,
- *povećane* (i *umanjene*), koji se od čistih razlikuju za pola stepena.

U Tabeli 2 dat je pregled intervala između tonova u oktavi.

Tabela 2: Pregled intervala između tonova

Interval	Veličina intervala u stepenima	Naziv
prima	0	čista
sekunda	$\frac{1}{2}$	mala
	1	velika
terca	$1\frac{1}{2}$	mala
	2	velika
kvarta	$2\frac{1}{2}$	čista

Interval	Veličina intervala u stepenima	Naziv
	3	prekomjerna
kvinta	3	umanjena
	$3\frac{1}{2}$	čista
seksta	4	mala
	$4\frac{1}{2}$	velika
septima	5	mala
	$5\frac{1}{2}$	velika
oktava	6	čista

Motiv je melodijska tema (ideja) koja služi kao osnova za dalji razvoj. *Muzička forma* je način organizovanja muzičke strukture na najvišem nivou. Ova organizacija može da bude striktno definisana (na primjer kod sonata) kada se pri komponovanju muzičkog djela pristupa „odozgo“, odnosno komponovanjem melodije tako da zadovolji unaprijed zadatu formu. Sa druge strane, mnoga muzička djela komponuju se „odozdo“, bez unaprijed zadate forme. Bez obzira da li je muzička forma unaprijed zadata ili ne, njena konačna struktura je najčešće hijerarhijska. Lako se može prepoznati da se muzičko djelo sastoji od sekcija ili *refrena*, a oni opet od muzičkih *fraza*.

Algoritamsko komponovanje muzike

Intenzivan razvoj informaciono komunikacionih tehnologija (IKT) utiče i na razvoj i trendove u ostalim granama ljudskog djelovanja, pa i u umjetnosti. Masovna produkcija elektronskog sadržaja raznih formata, (tekstualnih dokumenata, grafičkih objekata, video i audio zapisa, multimedijalnog materijala uopšte), zahtijeva intenzivan razvoj tehnika i alata za proizvodnju ovakvog sadržaja, kao i angažovanje velikog broja stručnjaka. Napredak tehnike i primjena informacionih tehnologija u svim sferama društva sa jedne strane omogućavaju, a sa druge i zahtijevaju dalji razvoj “tehničkog”, (polu) automatski generisanog materijala koji se dalje koristi kao osnovni ili pomoćni sadržaj. Ekranske pozadine, razne animacije, video materijal, grafike, tehničke slike kreirane u raznim kompjuterskim alatima za grafičku obradu, a i muzičke teme i semplovi su potrebni za realizaciju i poboljšanje radnog okruženja, osnovno i dodatno funkcionisanje tehničkih uređaja ili pak predstavljaju dodatne resurse za dalja istraživanja. Uz objektivan vizuelni ili čujni osjećaj koji korisnici ovakvih proizvoda doživljaju kao fizičku pojavu, neizostavno se javlja i subjektivan osjećaj ljepote.

Pored toga, svakodnevna primjena računara u u svim domenima ljudskog djelovanja dovodi i do pojave “kompjuterskog razmišljanja”. Stoga je danas moderno razgovarati o novim dostignućima, koristiti savremenu terminologiju, pratiti trendove razvoja IKT-a i primjenjivati ih u svakodnevnom životu. Ako se može reći da nauka ne smije podlijegati uticaju subjektivnog mišljenja, umjetnost je ta kod koje je moguće dostići punu slobodu u razmišljanju, interpretaciji i utisku. Primjena računara i drugih savremenih tehnologija se, bez obzira na oblast, u umjetnosti može smatrati dozvoljenim, čak i preporučljivim trendom. Bez obzira na objektivan kvalitet (kome uvijek treba težiti) koji današnja “virtuelna” ili automatski generisana “umjetnost”¹⁹ ima, ona je jedan od trendova kome teže pripadnici generacija koje su danas nosioci društva.

Sa druge strane, kreiranje “umjetničkih” djela pomoću kompjuterskih modela nailazi na opravdane i neopravdane kritike. Prije svega, “umjetnost” kreirana od strane računara ne može biti odraz autorovih emocija, raspoloženja ili namjera. Računari ne pokušavaju opisati nešto svojim “umjetničkim” djelima, jer naprosto nemaju emocije i subjektivni doživljaj [15]. Stoga, osoba kojoj se ovakve kreacije prezentuju ne može analizirati šta je autor (mašina) svojim djelom htio da kaže, već jedino na koji način je postignut rezultat. Ako se opet posmatra pretpostavka da je svaki kompjuterski program koji kreira neki materijal koji pretenduje da bude “umjetničkog karaktera” ipak osmislio čovjek, postavlja se pitanje da li je on (čovjek) zaista uspio (a prvo i želio) interpretirati svoju emociju ili raspoloženje, ili je krajnji proizvod više rezultat pravila nametnutih mašini. Takođe, ako se kreativnost smatra elementom umjetnosti, da li se pod kreativnošću može smatrati slučajna (slučajno generisana) promjena parametara koji učestvuju u kreiranju “umjetničkog” djela, koja se često sreće u raznim stohastičkim metodama. Dalje, potpunim determinizmom se ne postiže nikakva kreativnost. Upotreba elektronskih uređaja u umjetnosti uopšte, a u muzici posebno, već dovodi do problematičnih, a često i filozofskih razmatranja po pitanju šta se

¹⁹ ovdje se ne žele, bez dublje analize i rasprave, pod pojam umjetnosti svrstati razni eksperimentalni pokušaji automatskog kreiranja različitog materijala, bez obzira na eventualnu želju njihovih autora da takav materijal zaista i ima “umjetničku” crtu.

može, a šta ne može svrstati pod umjetnošću. Sama mogućnost i upotreba automatske sinteze različitih tonova inicira postavljanje pitanja određivanja granice između čovjekove kreativnosti i umjetničke sposobnosti sa jedne, te karakteristika mašine i automatske nekreativne proizvodnje tona sa druge strane.

S obzirom na prethodno, pitanje kvaliteta i smisla algoritamski generisanih “umjetničkih” dijela se nameće kao objektivno. Kada je o muzici konkretno riječ, protivnici algoritamskog komponovanja navode nekoliko objektivnih razloga koji minimizuju njegov značaj:

- muzika, kao umjetnost, ne može biti potpuno formalizovana i svedena na niz pravila na osnovu kojih se nepogrešivo komponuje,
- kvalitet proizvedog sadržaja često nije na zadovoljavajućem umjetničkom nivou, jer razvojni alati jednostavno nisu dobri,
- mnogi algoritmi su zasnovani na komponovanju manjih ili većih varijacija zadate teme, gdje se kreativnost, kao jedan od osnovnih elemenata umjetnosti zapravo pripisuje nekome drugom,
- često je u komponovanju ovakve muzike čovjek aktivno uključen (na primjer kod interaktivnih genetičkih algoritama), te se tako eventualna ljepota komponovanog djela opet ne pripisuje samom algoritmu, već čovjeku,
- mašine nemaju osjećaje, te je samim tim ovakva muzika besmislena, jer ne može biti interpretacija emotivnog stanja autora.

Nasuprot ovim objektivnim nedostacima algoritamskog komponovanja, niz činjenica ipak ide u prilog dosadašnjem, ali i daljem razvoju ovakve muzike:

- osjećaj ljepote je subjektivan, ne postoji opšte prihvaćena definicija dobre i loše muzike, tj. sve dok se nekome dati muzički proizvod sviđa, ima ga smisla i kreirati,
- neki elementi muzike su ipak “mjerljivi” i podliježu pravilima: frekvencije tonova, pravilnosti ritma, odnosi između skladnih tonova, jačina tona itd; svi ovi elementi se mogu formalizovati,
- kvalitet i kreativnost se postižu uzajamnim preplitanjem slučajnih (ali ne potpuno proizvoljnih) procesa i poboljšanjem trenutnog rješenja dok se ne odredi ono koje je zadovoljavajuće i za mašinu i za čovjeka,
- prema mišljenju kompozitora, proces (niz radnji prilikom) komponovanja djela od strane čovjeka se ne razlikuje od standardnog algoritamskog pristupa,
- pokušaji šabloniziranja se u istoriji muzike često javljaju. Kako se navodi u [6], u literaturi se pominje da su neki najčuveniji kompozitori (između ostalih i Bah, Mozart, Hajdn i Betoven) komponovanje pokušavali opisati skupom univerzalnih pravila, pozivajući se na svoje ili šablone drugih kompozitora,
- mnoge kompozicije dobijene na ovaj način i nemaju pretenziju da zauzmu mjesto umjetničkoj muzici koja je kroz vijekove komponovana od strane ljudskih bića; na primjer, kreiranje velikog broja eksperimentalnih semplova koji se koriste za sintezu ili dizajn zvuka je prostor koji je trenutno “nepopunjen” i novi zvukovi automatski generisani su dobrodošli,
- ljudski faktor odlučivanja, na kraju krajeva, nije izostavljen, jer je svaki kriterijum za komponovanje ipak sastavio čovjek.

U ovom radu se algoritamskom komponovanju (i konkretno upotrebi genetičkih algoritama u komponovanju) daje neutralno mišljenje. Algoritamska muzika koja se ovdje razmatra, a posebno rezultati dobijeni genetičkim algoritmima ilustruju upotrebu ovih algoritama u muzici, ali nemaju nikakvu namjeru da konkurišu komponovanju od strane čovjeka.

Načini algoritamskog komponovanja muzike

Prostor svih mogućih kompozicija nad definisanim skupom tonova je ogroman. Stoga se pretraživanje tog prostora, u cilju određivanja dovoljno dobrog rješenja, ne može vršiti pravolinijskim, prostim, strogo determinisanim algoritmima relativno niske složenosti. U pomoć nam dolaze složene metode pretraživanja i odlučivanja, koje najčešće povezujemo sa elementima vještačke inteligencije.

Ne postoji univerzalan način klasifikacije svih algoritama koji se koriste u komponovanju muzike. Ovo je uzrokovano različitim namjenama algoritama u komponovanju. Na primjer, klasifikacija može biti zasnovana na analizi na koji način algoritmi učestvuju u procesu komponovanja, tj. da li propisani algoritam sam komponuje muziku, ili algoritam samo pomaže u komponovanju. U prvom slučaju podrazumijeva se da se kompletna kompozicija kreira od strane algoritma (računara), dok se u drugom slučaju, na primjer, može smatrati da algoritmi samo odbacuju ili predlažu pojedina rješenja, vrše varijaciju teme, i sl, dok ostatak radi čovjek. Dalje, algoritmi se mogu klasifikovati na osnovu analize šta je krajnji rezultat algoritma, na primjer, da li je rezultat nekakav zapis kompozicije (na primjer notni ili MIDI²⁰ zapis), ili recimo, algoritam u realnom vremenu i komponuje i izvodi kompoziciju.

Pored ovih pristupa koji su zapravo više vezani za način primjene algoritama, za analizu algoritama u matematičkom smislu, algoritmi se klasifikuju po svojoj strukturi i načinu obrade podataka. Prema [15], algoritmi u komponovanju muzike se mogu klasifikovati na sljedeće grupe

- matematički modeli (eng. *mathematical models*),
- sistemi zasnovani na znanju (eng. *knowledge-based systems*),
- gramatike (eng. *grammars*),
- evolucionarni metodi (eng. *evolutionary methods*), kojima je posvećen ostatak rada,
- sistemi koji uče (eng. *systems which learn*),
- hibridni sistemi (eng. *hybrid systems*).

Matematički modeli

Matematički modeli su zasnovani na matematičkim jednačinama i slučajnim procesima. S obzirom na relativno nisku kompleksnost, ovi algoritmi su pogodni za upotrebu u aplikacijama koje zahtijevaju komponovanje u realnom vremenu (eng. *realtime applications*), te su tako prisutni u nekim

²⁰ Musical Instrument Digital Interface (MIDI) – standardni protokol za komunikaciju, kontrolu i sinhronizaciju zvuka, proizvedenog ili interpretiranog od strane nekog elektronskog uređaja – računara, klavijatura, i slično

komercijalnim softverskim rješenjima²¹. Kao često korištena stohastička metoda u literaturi se pominju Markovljevi lanci²².

Kao glavni nedostatak ovakvih algoritama, pored opštih nedostataka i problema algoritamskog komponovanja, prepoznaje se potreba za analizom velikog broja postojećih kompozicija, kako bi se na dobar način mogla odrediti vjerovatnoća za narednu slučajnu promjenljivu.

Sistemi zasnovani na znanju

Sistemi zasnovani na znanju podrazumijevaju sisteme koji su simbolički i koriste pravila ili ograničenja. Upotreba ovakvih sistema u muzici se čini prirodnim izborom, posebno kada pokušavamo odrediti prostor dozvoljenih melodija ili želimo uvesti eksplicitne strukture ili pravila. Osnovna prednost je ta što posjeduju mogućnost direktnog, eksplicitnog rezonovanja, tj. uvijek se može objasniti razlog za izbor odgovarajuće radnje. Ovakvi sistemi se, stoga mogu dosta praktično koristiti u harmonizaciji, jer se harmonizacija obično i smatra problemom koji treba da zadovolji propisana ograničenja (eng. *constraint satisfaction problem*). Značajne poteškoće u realizaciji ovakvih algoritama su vremenski zahtjevan i težak problem izdvajanja odgovarajućeg znanja, jer je problem identifikacije pravila, kao i izuzezaka u pravilima spojen i sa programerskim vještinama i vještinama u komponovanju.

Gramatike

Kompozicije se kreiraju tako što se prvo napravi „muzička gramatika“, koja se kasnije koristi za pravljenje muzičkih komada. Gramatike češće uključuju pravila na „makro nivou“ komponovanja, kao što su na primjer pravila u harmoniji ili ritmu, nego na pojedinim tonovima kompozicije. Stoga se ovakav pristup dosta koristi pri komponovanju džez muzike, koja podrazumijeva relativno jasna početna ograničenja uz visok stepen dozvoljene improvizacije.

Osnovni nedostaci ovakvih algoritama su

- gramatike su po pravilu hijerarhijske, dok sama muzika nije, te se u gramatike mora uvesti i višeznačnost, koja prepoznavanje dozvoljenih rješenja često čini kompjuterski zahtjevnim,
- većina ovakvih rješenja ne daje značaja samom značenju dobijenog komada. Gramatike su tu da daju pravila i ograniče skup dozvoljenih rješenja, dok se ne analizira puno šta se dobija primjenom tih pravila. Stoga gramatike generišu veliki broj muzičkih zapisa od kojih su mnogi lošeg kvaliteta (gramatike odbacuju samo neke nepravline i loše kompozicije).

Sistemi koji uče

Kod sistema koji uče pretpostavljamo da unaprijed nemamo nikakvo znanje (pravila za pravljenje kompozicija, ograničenja i slično), već sistem sam uči preko primjera. Ovi sistemi se mogu klasifikovati po načinu čuvanja informacija (stečenog znanja) na distributivne (Vještačke neuronske mreže) i simboličke (Mašinsko učenje). Ideja upotrebe ovakvih algoritama je mogućnost automatskog učenja prepoznavanja

²¹ Na primjer, Jam Factory, softversko rješenje realizovano od strane američke kompanije Intelligent Music, New York

²² Markovljevi lanci su niz slučajnih promjenljivih kod kojih raspodjela vjerovatnoća za buduće stanje zavisi samo od trenutnog, dok je u odnosu na sva prethodna stanja nezavisno.

kvaliteta kompozicije i pravljenje inteligentnih odluka. Teoretski, ovi sistemi imaju prednost u odnosu na sisteme zasnovane na znanju u tome što mogu da uče na osnovu primjera. Ovo bi potencijalno moglo dati rezultate pri oponašanju ljudske kreativnosti, jer je sama kreativnost često zasnovana na ukidanju ili nepoštovanju pravila. Sistemi koji uče su usko povezani sa teorijom vjerovatnoće i statistikom i tzv. *data mining*-om²³, pomoću kog se vrši duboka analiza raspoloživih podataka, kako bi se na osnovu nje izvukli zaključci o strukturi i potencijalu samih podataka.

Kao osnovni nedostatak ovog pristupa prepoznaje se nemogućnost da algoritam „dovoljno dobro nauči“ sve elemente koje dobra kompozicija sadrži, te je sam proces komponovanja i kvalitet dobijenih kompozicija diskutabilan. Pored toga, algoritmi su vremenski izuzetno zahtjevni, te su i neprimjenljivi za komponovanje u realnom vremenu.

Hibridni sistemi

Hibridni sistemi predstavljaju kombinaciju različitih tehnika vještačke inteligencije. Objedinjavanje prednosti pojedinačnih pristupa opravdava upotrebu ovakvih sistema. Na primjer, jednim sistemom (recimo genetičkim algoritmom) se određuje ritam, dok se funkcija koja ocjenjuje kvalitet same melodije (tonova) računa upotrebom vještačkih neuronskih mreža.

Osnovni nedostaci su komplikovanost same implementacije i visoka vremenska zavisnost.

Bez obzira na silna ograničenja kojih se većina kompozitora pridržava u procesu komponovanja, a koja su prenesena na različite sisteme algoritamskog komponovanja, prostor “dozvoljenih” tonova i njihovih kombinacija je ipak još uvijek prevelik da bi se sa bilo kakvom sigurnošću neki algoritam za komponovanje muzike mogao smatrati pouzdanim. Ogroman broj kombinacija tonova²⁴, uz pretpostavku da je i njihovo trajanje proizvoljno, garantuje da će prostor muzičkih djela ostati neistražen, bez obzira na to koliko dobar algoritam bio i koliko se brzo izvršavao. Upotreba savremenih računara i moćnih algoritama u komponovanju pomaže u proučavanju muzike i povećava mogućnosti njene upotrebe, ali sa druge strane, nemogućnost mašine da procijeni šta je lijepo, a šta ne, upravo pokazuje po čemu se umjetnost razlikuje od ostalih ljudskih djelatnosti.

²³ *data mining* podrazumijeva duboku analizu podataka, kako bi se iz sadržaja i prirode tih podataka izvukle dodatne, skrivene informacije

²⁴ Matematički gledano, ovaj skup je beskonačan i prebrojiv.

Genetički algoritmi u muzici

Genetički algoritmi

Genetički²⁵ algoritmi su složeni i prilagodljivi algoritmi koji omogućavaju rješavanje nekih robusnih problema optimizacije. U osnovi, podrazumijeva se rad sa populacijom jedinki, gdje svaka jedinka predstavlja potencijalno (optimalno) rješenje, a svaka populacija je podskup ukupnog prostora pretraživanja. Populacija se u iterativnom postupku mijenja (stare jedinke se mijenjaju novim – boljim).

Svakoj jedinki se dodjeljuje vrijednost tzv. fitnes - prilagođenost (eng. *fitness*) funkcije, koja ocjenjuje kvalitet posmatrane jedinke. Cilj genetičkog algoritma je da se iz iteracije u iteraciju pronalaze jedinke sa sve boljim fitnesom, pod čime se smatra i pojedinačno poboljšanje svake jedinke, ali i srednja prilagođenost kompletne populacije, što se postiže standardnim genetičkim operatorima: selekcijom, ukrštanjem i mutacijom.

Selekcija favorizuje kvalitetne (natprosječne) jedinke (one sa većim fitnesom), kao i njihove nadprosječne dijelove (gene), koji dobijaju veću šansu da „prežive“ u sljedeću generaciju i da se međusobno reprodukuju prilikom formiranja nove generacije. Slabije prilagođene jedinke i geni imaju smanjene šanse za reprodukciju, te postepeno izumiru.

Ukrštanjem se rekombinuju geni jedinki. Rezultat ukrštanja jedinki su nove jedinke, koje potencijalno sadrže „dobre“ gene jedinki od kojih su nastale. Ovim mehanizmom se i jedinke slabijeg fitnesa, ali koje sadrže dobre gene dobijaju šansu da učestvuju u reprodukciji i prenesu „dobar“ genetski materijal na sljedeću generaciju.

Mutacijom se vrši slučajna promjena određenog gena (sa nekom malom vjerovatnoćom da će se mutacija desiti), čime se postiže mogućnost vraćanja dobrog genetskog materijala ukoliko je on izgubljen višestrukom primjenom selekcije i ukrštanja.

Svaka jedinka se predstavlja genetskim kodom nad određenom konačnom azbukom. U širokoj primjeni genetičkih algoritama, uobičajeno je da se koristi binarno kodiranje, gdje se genetski kod sastoji od niza bitova. Populaciju najčešće čini između 10 i 200 jedinki.

Početna populacija se uglavnom generiše na slučajan način, što doprinosi raznovrsnosti genetskog materijala. Često se za generisanje početne populacije koristi neka druga heuristika, , gdje je jedini preduslov da izvršenje te druge metode bude relativno kratko.

Kod najprostije varijante genetičkog algoritma selekcija se zasniva na prostoju rulet selekciji, jednopozicionom ukrštanju i prostoju mutaciji. Ovakav genetički algoritam se u literaturi obično naziva prost genetički algoritam (eng. *simple genetic algorithm* – SGA). Detaljan opis navedenih operatora dat je u narednom poglavlju. Na Slici 2 prikazan je sistem osnovnog funkcionisanja genetskog algoritma. [16]

²⁵ U literaturi na našem jeziku se koriste izrazi genetski ili genetički algoritmi, a obje riječi su prevod engleske riječi *genetic*.

```

Učitavanje_Ulaznih_Podataka();
Inicijalizacija_Populacije();
while (! Kriterijum_Završetka_GA() )
{
    for (i=0; i< Npop; i++) pi = Vrednosna_Funkcija();
    Funkcija_Prilagođenosti();
    Selekcija();
    Ukrštanje();
    Mutacija();
}
Štampanje_Izlaznih_Podataka();

```

Slika 2: Osnovna šema genetičkog algoritma

Osnovni dijelovi implementacije genetičkih algoritama, sa aspektima upotrebe u muzici

Najvažniji aspekti svakog genetičkog algoritma su: način kodiranja jedinki, fitnes funkcija, način zamjene generacije i operatori: selekcija, mutacija i ukrštanje.

Kodiranje

Najčešći način kodiranja kod genetičkih algoritama je binarno kodiranje, gdje se genetski kod sastoji od niza bitova. Prilikom rješavanja složenih optimizacionih problema moguća su i druga kodiranja (podrazumijeva se sa azbukom veće kardinalnosti), ali su mišljenja o takvom kodiranju podijeljena. Binarno kodiranje, sa druge strane, omogućava efikasnu implementaciju genetskih operatora, a s obzirom na to da se kod ovih optimizacionih metoda dosta pažnje posvećuje brzini dostizanja optimalnog rješenja, binarno kodiranje se u većini slučajeva čini optimalnim.

U komponovanju muzike pomoću genetičkih algoritama ne postoji univerzalno usvojen način kodiranja tonova i kompozicije. Razlog za ovo leži u prilično različitim pristupima komponovanju: na primjer, da li se kompozicija sastoji samo od niza pojedinačnih tonova, ili su dozvoljeni i akordi, dalje, koliko je tonova u opticaju, da li se ritam uopšte razmatra ili ne, i slično. Način predstavljanja takođe može zavistiti i od načina interpretacije dobijene melodije, na primjer, da li se melodija komponuje i interpretira u realnom vremenu, ili se dobijene kompozicije samo arhiviraju u odgovarajući zapis.

Prostor koji se pretražuje (tonovi, odnosno nizovi tonova) je diskretnog karaktera, te upotreba binarnog kodiranja nekada može biti nepraktična. Kako se tonovi smatraju pojedinačnim jedinkama, njihovo predstavljanje pomoću binarnog zapisa bi kasnije moglo dovesti do nerješivih problema (na primjer, šta bi predstavljala mutacija jednog bita u jednom nizu tonova, a mogli bi nastati i problemi kod ukrštanja dvije jedinke). Kao dodatna dimenzija svake jedinke javlja se i ritam, pa se za predstavljanje jedinke treba uvesti i druga dimenzija (pored visine tonova), koja se odnosi i na trajanje tona.

Pri predstavljanju mora se voditi računa o tome da svaka kompozicija koja dolazi u obzir mora imati jednu i samo jednu reprezentaciju i da reprezentacija treba da bude praktična za kasniju upotrebu u algoritmu. Takođe, reprezentacija treba da bude takva da se od nje ponovo može napraviti (u muzičkom smislu, na primjer, kao audio ili notni zapis) odgovarajuća kompozicija, bez ikakvog gubitka ili dodavanja informacija.

U praksi se prepoznaju dva aspekta kodiranja [14]

- Kodiranje pojedinačnih tonova i njihovog trajanja,
- Kodiranje nizova (tonova) koji čine (dio ili čitavu) kompoziciju.

Kodiranje pojedinačnih tonova

Karakteristike tona koje su značajne za genetičke algoritme su visina i trajanje.

Postoje tri načina predstavljanja visine tona: apsolutno, relativno i pomoću „skalarnog odstupanja“²⁶.

Apsolutna reprezentacija podrazumijeva jednoznačno mapiranje svakog elementa na odgovarajuću skalu tonova, predstavljenu ili standardnim imenima (C3,F#5), MIDI kodom (0-127) ili odgovarajućim frekvencijama (na primjer, ton A4 ima frekvenciju 440 Hz). Relativna reprezentacija podrazumijeva predstavljanje tonova kao intervale u odnosu na neki referentni ton. Skalarno odstupanje predstavlja kombinaciju prethodna dva načina, gdje se svaki ton predstavlja preko odstupanja u odnosu na trenutni akord koji je u osnovi promjenljive harmonije.

Sva tri pristupa imaju svoje prednosti. Zbog jednoznačnosti predstavljanja tonova, apsolutna reprezentacija je najjednostavnija, dok se relativna reprezentacija koristi ukoliko se kompozicija želi transponovati iz jednog tonaliteta u drugi. Kod skalarnog odstupanja nazivi nota su uvijek u skladu sa trenutnom harmonijom, ali je razlika u tome što se harmonija može mijenjati.

Postoje dva načina predstavljanja trajanja tonova: Reprezentacija zasnovana na otkucajima (eng. *beat oriented*) i apsolutno trajanje. Reprezentacija zasnovana na otkucajima podrazumijeva upotrebu uobičajene terminologije i notacije za trajanje tonova (četvrtina, osmina, i sl). Apsolutna reprezentacija podrazumijeva upotrebu vremenskih konstanti koje se najčešće izražavaju u milisekundama. Reprezentacija zasnovana na otkucajima se koristi u kompozicijama gdje je ritam značajan element, posebno u slučajevima kada se predviđa i promjena tempa. Apsolutna reprezentacija se koristi u slučajevima kada dužina trajanja tonova nije ograničena na unaprijed definisan skup dozvoljenih vrijednosti trajanja.

Kodiranje nizova tonova

Značajni slučajevi kodiranja nizova tonova su tzv. predstavljanje zasnovano na poziciji (eng. *position based*) i predstavljanje zasnovano na poretku (eng. *order based*).

Predstavljanje zasnovano na poziciji počiva na sljedećem principu [17]: Izabere se neka mala (relativna ili apsolutna) vremenska konstanta koja je, matematički gledano, najveći zajednički djelilac svih mogućih

²⁶ eng. *scale offset*, prevod nije idealan

dužina trajanja, (najčešće je to najkraća dozvoljena dužina trajanja tona). Tada se dužina svakog tona predstavlja kao odgovarajući umnožak izabrane male dužine, dok se sam ton predstavlja vezanjem odgovarajućeg broja kratkih tonova iste visine. Na primjer, ako je riječ o tročetvrtinskom taktu, a vremenska konstanta je šesnaestina tona, tada se čitav takt dijeli na $4 \cdot 3 = 12$ šesnaestina, osmina otkucaja se predstavlja spajanjem dvije šesnaestine, četvrtina spajanjem četiri, polovina pomoću osam itd.

Kao rezultat ovakvog pristupa, praktično gledano, svaki hromozom (niz gena) – jedinka- je fiksne dužine, a svaki gen daje informaciju ili o samom tonu, ili o produženom trajanju prethodnog tona.

Predstavljanje zasnovano na poretku se realizuje na osnovu kreiranja odgovarajućih parova visina-trajanje. Ova reprezentacija se koristi kada je potrebno razdvojiti strukture u kojima se čuvaju visina i trajanje (u prethodnom slučaju sve se čuva na jednom mjestu), a podesna je za upotrebu gdje nije unaprijed definisan skup trajanja tonova, ili je taj skup jako veliki.

Primjer predstavljanja kompozicije pomoću relativnog predstavljanja tonova

Razmotrimo prvo opšti slučaj.

Neka je skup dozvoljenih tonova podskup standardnog dijatonskog skupa tonova²⁷.

Neka je izabran sistem relativnog predstavljanja tonova i neka je ukupan broj (visina) tonova n . Neka je referentnoj visini tona dodijeljen broj 1, sljedećem (po visini) tonu broj 2, narednom 3 itd. Dalje, neka je najveći zajednički djelilac svih trajanja broj k . Tom trajanju dodijelimo oznaku „jedinična dužina“. Neka se čitava kompozicija sastoji od m taktova, a svaki takt od po p otkucaja. Neka u jednom otkucaju ima q jediničnih dužina. Odavde zaključujemo i konstatujemo:

1. svaki takt je trajanja pq jediničnih dužina
2. svaka dužina tona je trajanja tq jediničnih dužina, za neko t .
3. čitava kompozicija je trajanja mpq jediničnih dužina
4. pauza jedinične dužine se označava brojem 0
5. jedinična dužina se predstavlja brojem $n+1$.
6. za predstavljanje čitave kompozicije dovoljan nam je niz (brojeva) ukupne dužine mpq , gdje su svi elementi tog niza iz intervala $[0, n+1]$. Ako je element niza iz intervala $[1, n]$, riječ je tonu odgovarajuće visine, ako je element nula, riječ je o pauzi, a ako je element jednak $n+1$, to znači da je prvom tonu (ili pauzi) sa lijeve strane trajanje produženo za najkraću dužinu.
7. svakoj kompoziciji koja zadovoljava početne uslove može se dodijeliti jedan i samo jedan niz.
8. Svakom nizu, osim onih koji počinju brojem $n+1$ (ne zna se šta je jedinične dužine) odgovara tačno jedna kompozicija.

Ovakvim sistemom se postiže relativno jednostavna reprezentacija jednostavnijih kompozicija, dok se za one složenije ovaj sistem može koristiti uz dodatna proširenja. Na primjer, ova osnovna postavka ne dozvoljava predstavljanje više tonova u jednom trenutku, te se praktično, za svaku

²⁷ Na primjer, skup tonova je takav da se svaki ton može odsvirati odgovarajućom dirkom na klaviru

takvu situaciju mora uzeti više nizova. Dalje, ovakvim sistemom, iako teoretski moguće, nije praktično predstavljanje poliritmičnih kompozicija, tj. onih koje imaju širok dijapazon različitih trajanja (na primjer, ako su, pored uobičajenih trajanja četvrtine, osmine, šesnaestine, prisutna i trajanja od trećine, petine, šestine otkućaja).

Pogledajmo kako bi se ovakvo predstavljanje primijenilo na konkretnu kompoziciju. Na Slici 3 je data kompozicija²⁸ predstavljena notnim zapisom i odgovarajućim kodiranjem.



Slika 3: Primjer kodiranja tonova

Neka je, s obzirom da je kompozicija u a molu, ton C4 izabran za referentni. Neka je ukupno na raspolaganju dvije oktave. Tonovi koji ne pripadaju skali C dura²⁹ (odnosno a mola) se (u ovom primjeru) ne razmatraju, te za njih nema odgovarajuće reprezentacije³⁰. Brojevi iznad nota označavaju udaljenost od referentnog tona.

S obzirom na to da ukupno imamo 14 različitih tonova, možemo se odlučiti za reprezentaciju prikazanu na Slici 4. Vidimo da je brojem nula označena pauza.



Slika 4: Kodiranje tonova po C duru

Sada je potrebno uvesti trajanje tonova. Na osnovu referentne kompozicije možemo primijetiti sljedeće činjenice: Kompozicija je zapisana u četvoročetvrtinskom taktu. Pošto je ukupan broj dozvoljenih tonova 14, svi elementi niza su iz intervala [0,15], gdje se nulom označava pauza, a brojem 15 dodavanje jedne najkraće dužine na trajanje prethodne note. Najveći zajednički djelilac svih trajanja (najkraća dužina) je jedna osmina. Stoga za najkraću dužinu uzimamo osminu otkućaja. S obzirom na to da kompozicija ukupno ima četiri takta, a u svakom taktu imamo osam najkraćih dijelova, za predstavljanje ove kompozicije potreban nam je niz dužine 32. Pauza (i to ona trajanja jednu osminu) se predstavlja nulom,

²⁸ Posmatrana kompozicija je tema poznate popularne kompozicije sarajevskog muzičara Harija Varešanovića

²⁹ Ovo je dovoljno povezati sa bijelim tipkama na klaviru

³⁰ Ovo ne umanjuje opštost, jer se samo jednostavnim proširenjem skupa dozvoljenih tonova (a samim tim i odgovarajućeg niza) može dobiti reprezentacija svakog polutona

svaki ton se predstavlja odgovarajućim brojem koji predstavlja trajanje od jedne osmine. Svako duže trajanje se označava brojem 15. Tako prvi ton C koji se javlja u kompoziciji traje tri osmine, te ga predstavljamo pomoću 8 15 15. čitav niz izgleda kao u Tabeli 3:

Tabela 3: Prikaz kodiranja čitave kompozicije prikazane na Slici 3.

Elementi niza i odgovarajuće vrijednosti																															
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
0	3	6	7	8	15	15	7	8	7	6	5	4	15	15	15	0	4	5	6	7	15	15	6	7	6	5	4	3	15	15	15

Raspored odgovarajućih brojeva u odnosu na note predstavljen je na šemi Slika 5.



Slika 5: Kodiranje kompozicije sa zapisivanjem pauza i trajanja

Fitnes funkcija

Fitnes funkcijom određujemo kriterijum za poređenje kvaliteta jedinki. Problem određivanja fitnes funkcije u teoriji genetičkih algoritama je često kritična tačka u samom projektovanju algoritma. Kada se u obzir uzme i dodatno opterećenje da je muzika subjektivan čulni događaj (ono što se dopada jednoj osobi ne mora biti prijatno i drugoj – pa prema tome, kako god se računao fitnes, ostaje mogućnost subjektivnog mišljenja o kvalitetu jedinke), jasno je da je određivanje fitnes funkcije genetičkog algoritma najvažniji, ali i najkomplikovaniji pojedinačni korak. Prema trenutnom stanju u umjetnosti, još uvijek nije definisan pouzdan i efikasan način određivanja fitnes funkcija koje će direktno uputiti ka željenom rješenju [18]. U većini slučajeva, kada je riječ u automatskom određivanju fitnesa, koristi se multiobjektivna funkcija (funkcija koja računa ukupan fitnes na osnovu više različitih kriterijuma).

Stoga ne iznenađuje da se u literaturi naglasak često stavlja na definisanje fitnes funkcije: na koji način selektovati najbolju jedinku unutar populacije? Na koji način definisati „dobru“ muziku?

Postoje dva osnovna načina mjerenja fitnesa jedinke.

1. Automatsko određivanje fitnesa (eng. *automatic fitness assessment* – AFA). Podrazumijeva kodiranje dovoljne količine znanja i definisanje dovoljnog skupa kriterijuma na osnovu kojih se vrši računanje fitnesa jedinki i selekcija sljedeće generacije.
2. Upotreba ljudske kritike za pravljenje izbora. Ovdje se susrećemo sa interaktivnim genetičkim algoritmima (eng. *interactive genetic algorithm* – IGA), kod kojih se procjena kvaliteta dobijene melodije ostavlja čovjeku, koji tako kontroliše odvijanje algoritma i „presuđuje“ koja je jedinka dobra i kolikoj mjeri.

Automatsko određivanje fitnesa

Prepoznaju se tri vrste automatskih fitnes šema:

- šema zasnovana na heuristici (eng. *heuristic features*),
- šema zasnovana na pravilima (eng. *rule based*)
- šema zasnovana na učenju (eng. *learned*)

Heurističke šeme uključuju određivanje fitnesa zasnovanog na ispitivanju opsega i vrsta tonova ili intervala, smjeru i stabilnosti melodije, odnosu tonova i pauza, ritmičkim elementima, sinkopama, kao i drugim funkcijama koje mogu vratiti neku brojevu ili logičku vrijednost. Ukupan fitnes se na kraju računa kao suma vrijednosti pojedinačnih funkcija, prethodno pomnoženim odgovarajućim težinskim faktorom, koji zavisi od toga kojoj osobini melodije se daje veći značaj.

Osnovni problem koji se javlja prilikom upotrebe ovog sistema ocjenjivanja kvaliteta dobijene melodije je problem određivanja skupa ograničenja (pojedinačnih funkcija), kao i „težine“ odgovarajućeg ograničenja. Pokušaji da se na jednostavan način (korištenjem jednog ili dva kriterijuma) odredi kvalitet dobijene melodije nisu urodili plodom [15]. Sa druge strane, veliki broj kriterijuma koji utiču na ukupan kvalitet (fitnes) često unosi konfuziju za određenje fitnesa jer se može desiti da dva različita kriterijuma različito vrednuju jednu istu osobinu kompozicije. Stoga se, na primjer, u [19] konstatuje da je ovakav način vrednovanja kompozicija upitne vjerodostojnosti.

Šeme zasnovane na pravilima implementiraju tzv. sisteme zasnovane na znanju (eng. *knowledge - based systems*). Obično, pravila na osnovu kojih se komponuje predstavljaju odgovarajuća ograničenja. Ovakva ograničenja u principu mogu da eliminišu eventualna loša rješenja, ali nisu sposobna da gradiraju ona rješenja koja su teorijski korektna. Tako imamo situaciju da neki zvukovi (ili kompozicije) koje su teoretski ispravne (zadovoljavaju sva ograničenja) ipak zvuče loše, dok se dobre kompozicije zbog kršenja nekih pravila eliminišu. Ovdje zapravo dolazimo do klasične greške u upotrebi genetičkih algoritama, a to je da funkcija fitnesa ne odražava kvalitet jedinke.

Šeme zasnovane na učenju koriste tehnike vještačke inteligencije kako bi se suzio skup pretrage na osnovu učenja iz skupa prihvatljivih primjera. Dominantna tehnika koja koristi ovaj princip su vještače neuronske mreže (eng. *artificial neural networks*). Vještače neuronske mreže (ili samo neuronske mreže) su matematički ili računarski modeli zasnovani na modelu bioloških neuronskih mreža. Sastoje se od međusobno povezanih (vještačkih) *neurona*³¹ i sposobne su da procesiraju informacije između njih. Neuronske mreže su adaptivni sistemi koji mijenjaju svoju strukturu na osnovu spoljašnjih ili unutrašnjih informacija koje teku kroz mrežu tokom faze učenja.

„Mjerljivi“ elementi kompozicije

Po pravilu, fitnes ima numeričku vrijednost (ili vrijednost koja ima svoju numeričku interpretaciju). Da bi se odredio fitnes jedinke (kompozicije), potrebno je odrediti koje osobine kompozicije uopšte možemo mjeriti i predstaviti brojem. U [19] je data detaljna lista „mjerljivih“ elemenata kompozicije. Svaka

³¹ Pod *neuronom* se smatra objekat, odnosno funkcija koja može da primi skup podataka i na osnovu njih izračuna izlazni rezultat koji prosljeđuje drugim *neuronima*.

osobina se mjeri odnosom odgovarajućih vrijednosti. Najkraća dužina³² se definiše kao najveći zajednički djelitelj svih trajanja. Prepoznate su ukupno 21 osobina koje su klasifikovane u grupe osobina sa sljedećim cjelinama:

- Visina tonova,
- Osobine tonaliteta,
- Nizovi rastućih/opadajućih tonova – konture tonova,
- Ritmički elementi,
- Šabloni (paterni).

Visina tonova

Raznovrsnost tonova. Definiše se kao odnos ukupnog broja upotrijebljenih različitih visina tonova i ukupnog broja svih tonova.

Opseg tonova. Definiše se kao odnos razlike između visina najvišeg i najnižeg tona (najčešće kao broj polutonova između te dvije visine) i ukupnog broja raspoloživih tonova.

Osobine tonaliteta

Centriranost u odnosu na tonalitet. Definiše se kao odnos zbira svih najkraćih trajanja tonova koji odgovaraju ili osnovnom tonu tonaliteta ili njegovoj čistoj kvinti i ukupnog zbira svih najkraćih trajanja.

Tonovi koji ne pripadaju skali izabranog tonaliteta. Definiše se kao odnos zbira najkraćih trajanja svih tonova koji ne pripadaju datoj skali i ukupnog zbira svih najkraćih trajanja.

Disonantni intervali. Definiše se kao odnos broja svih disonantnih intervala i ukupnog broja intervala³³. U literaturi se sreće različito vrednovanje intervala. Po pravilu, savršenim konsonantima (primi, čistoj kvarti, čistoj kvinti i oktavi) se daju veće vrijednosti, nesavršenim konsonantima: maloj i velikoj terci, te maloj i velikoj seksti srednje, dok se maloj i velikoj sekundi, odnosno maloj i velikoj septimi daju najmanje vrijednosti.

Nizovi rastućih/opadajućih tonova – konture tonova

Rastući intervali. Definiše se kao odnos broja svih rastućih intervala i ukupnog broja svih intervala. Provjerava se da li melodija ima tendenciju rasta ili opadanja tonova.

Stabilnost konture³⁴. Definiše se kao odnos broja intervala kod kojih sljedeći interval ide u istom smjeru i ukupnog broja svih intervala umanjenog za 1.

Kretanje za po (polu)ton. Definiše se kao odnos broja intervala koji su mala (ili velika) sekunda i ukupnog broja intervala.

³² U pomenutom članku, najkraćoj dužini je dodijeljen engleski termin *quantum*. U kasnijem tekstu koristi se termin najkraća dužina

³³ Ukupan broj intervala iznosi $ukupanBrojNota - 1$

³⁴ Pod konturom se podrazumijeva rastući ili opadajući niz tonova

Nagli skokovi. Definiše se kao odnos broja „velikih“ intervala između jednog i drugog tona (na primjer, najmanje mala seksta) i ukupnog broja intervala, ali pod uslovom da naredni (treći) ton nije isti kao i prvi.

Snaga vrhunca³⁵. Definiše se kao recipročna vrijednost broja nota koje predstavljaju vrhunac kompozicije.

Ritmički elementi

Gustoća nota. Definiše se kao odnos ukupnog broja nota i ukupnog broja najkraćih dužina.

Gustoća pauza. Definiše se kao odnos ukupnog broja pauza i ukupnog broja najkraćih dužina.

Raznovrsnost ritma. Definiše se kao odnos ukupnog broja različitih trajanja nota i ukupnog broja svih mogućih trajanja (na primjer, svih uobičajenih trajanja tonova od tridesetdruginke do cijele note).

Opseg trajanja. Definiše se kao odnos količnika najdužeg i najkraćeg trajanja i ukupnog broja svih mogućih trajanja.

Sinkope. Definiše se kao odnos broja sinkopa³⁶ i ukupnog broja nota.

Šabloni (paterni)

Ponavljanje tonova. Definiše se kao odnos broja intervala koji su prime i ukupnog broja intervala.

Ponavljanje ritmičkih vrijednosti. Definiše se kao odnos broja intervala kod kojih su obje note intervala istog trajanja i ukupnog broja intervala.

Ponavljanje šablona od tri note, po visini tonova. Definiše se kao odnos ukupnog broja nizova od tri uzastopne note koje se (po visini) ponavljaju i ukupnog broja nota umanjenog za 4. Pauze se takođe smatraju dijelom šablona.

Ponavljanje šablona od četiri note, po visini tonova. Definiše se kao odnos ukupnog broja nizova od četiri uzastopne note koje se (po visini) ponavljaju i ukupnog broja nota umanjenog za 5. Pauze se takođe smatraju dijelom šablona.

Ponavljanje šablona od tri note, po trajanju tonova. Definiše se kao odnos ukupnog broja nizova od tri uzastopne note koje se (po dužini trajanja) ponavljaju i ukupnog broja nota umanjenog za 4. Pauze se takođe smatraju dijelom šablona.

Kriterijumi za fitnes funkciju

Na osnovu nabrojanih „mjerljivih“ osobina, mogu se prepoznati neki kriterijumi koji mogu učestvovati u računanju ukupnog fitnesa. Početne postavke u algoritmu definišu koje osobine i u kojoj mjeri utiču na kvalitet kompozicije. Data osobina se može direktno posmatrati kao čitav fitnes, ili, što je češći slučaj,

³⁵ Prevod sintagme *climax strength* – subjektivni ili objektivni osjećaj vrhunca kompozicije

³⁶ Pod sinkopom se podrazumijeva pojava kada je trajanje note veće ili jednako trajanju otkućaja, a početak trajanja te note se ne poklapa sa početkom otkućaja.

vrijednost fitnes funkcije se dobija kao linearna kombinacija više vrijednosti koje predstavljaju odraz nekih mjerljivih osobina. Dakle, fitnes f neke kompozicije se definiše kao

$$f = \sum_{i=1}^n \lambda_i f_i$$

gdje λ_i predstavlja težinu (uticaj) vrijednosti f_i na čitav fitnes, a n je ukupan broj različitih kriterijuma. Na primjer, za različite i , f_i mogu biti odnos između broja tonova van datog tonaliteta i ukupnog broja tonova, odnos broja neskladnih intervala i svih intervala, odnos broja (ili ukupan broj) pojavljivanja nekih šablona u odnosu na ukupan broj nota, gustoća nota i slično. Odgovarajuće *lambdae* daju težinu dogovarajućoj vrijednosti.

Dalje, fitnes se može računati i po taktovima, gdje ukupan fitnes tada predstavlja zbir svih fitnesa po taktovima. U najopštijem slučaju, fitnes možemo definisati kao

$$f = \sum_{j=1}^k \sum_{i=1}^n \lambda_{ij} f_i$$

gdje λ_{ij} predstavlja uticaj vrijednosti f_i u j -tom taktu, gdje je n ukupan broj kriterijuma, a k broj taktova.

Interaktivno određivanje fitnesa

Interaktivni fitnes koristi čovjeka kao mentora, koji svakoj pojedinačnoj jedinki procjenjuje kvalitet.

Prednosti ovakvog određivanja fitnesa se očigledno odnose na izbjegavanje automatskog određivanja kvaliteta dobijene kompozicije, što je, kako je već rečeno, u najmanju ruku problematično.

Osnovni nedostatak interaktivnih genetičkih algoritama je tzv. pojava uskog grla (eng. *bottleneck*) koje nastaje usljed toga što čovjek mora pažljivo poslušati svaku jedinku populacije kako bi joj odredio fitnes. Ovo je značajan ograničavajući faktor, kako zbog ograničenja veličine populacije, tako i ukupnog broja generacija[2]. Sa druge strane, ovakvim pristupom u mnogome je izbjegnuta najveći problem algoritamskog komponovanja, automatsko računanje kvaliteta dobijene melodije.

Genetički operatori

Pored načina određivanja fitnesa, izbor operatora dosta utiče i na sam proces izvršenja algoritama, kao i na mogućnost dobijanja dovoljno dobrog rješenja.

Postoji pet standardnih operatora u genetičkim algoritmima. To su inicijalizacija, selekcija, ukrštanje, mutacija i zamjena. U rješavanju konkretnih, specifičnih problema, kao što komponovanje muzike, ovi operatori se često koriste drugačije od uobičajenih načina. Ovdje se prvenstveno misli na operatore ukrštanja i mutacije. [16]

Inicijalizacija

Postoje dva osnovna načina kreiranja inicijalne populacije:

- slučajno izabrana populacija,

- populacija zasnovana na semplovima.

Slučajno izabrana populacija podrazumijeva odabir početne populacije na potpuno slučajan način. Izbor početne populacije u opštem slučaju ne treba da utiče na konvergenciju ka optimalnom rješenju. Ovakvo generisanje početne populacije je najčešće zasnovano na upotrebi standardnog kompjuterskog generatora slučajnih brojeva. Iako je ovakav način generisanja početne populacije u genetičkim algoritmima uobičajen, u muzici često rezultuje potpuno amuzičnom početnom populacijom. Ova situacija može da predstavlja problem u određivanju fitnesa pojedinačnih, loših jedinki, posebno u slučaju interaktivnih genetičkih algoritama. Stoga se umjesto potpuno slučajnog generisanja početne populacije može koristiti neka populacija dobijena drugim metodama, kao na primjer upotrebom fraktalnog generatora ili Markovog lanca.

Kod upotrebe semplovane inicijalne populacije pretpostavlja se određivanje početne populacije koja sadrži već prihvatljive jedinke. Cilj upotrebe ovakve početne populacije je izbjegavanje velikog broja nemuzičnih skupova tonova koji mogu nastati potpunim slučajnim generisanjem. Dalje, upotreba ovakve početne populacije uglavnom se više odnosi na pravljenje varijacija i/ili kombinacija tih početnih melodija (semplova) nego na čisto komponovanje.

Selekcija

Selekcija predstavlja način odabira sljedeće generacije na osnovu jedinki dobijenih ukrštanjem i mutacijom prethodne. Postoje tri osnovna načina selekcije:

- standardna selekcija zasnovana na fitnesu (eng. *fitness based selection*),
- selekcija zasnovana na muzičkim elementima populacije (jedinki) (eng. *music aware selection*),
- slučajna selekcija (eng. *random selection*).

Standardna selekcija zasnovana na fitnesu podrazumijeva izbor jedinki iz trenutne generacije po principu: bolji fitnes, veća vjerovatnoća da će jedinka biti izabrana. Ovdje može biti primjenjen neki od sljedećih principa:

- tzv. rulet selekcija, kod koje je vjerovatnoća da će jedinka preživjeti za sljedeću generaciju direktno proporcionalna njenom fitnesu,
- elitistička strategija, gdje se određen broj najboljih jedinki direktno prosljeđuje u sljedeću generaciju,
- turnirska selekcija, gdje se (u osnovnoj varijanti) vrši poređenje (po fitnesu) po dvije proizvoljne jedinke, a u sljedeću generaciju se bira bolja (ona sa većim fitnesom).
- eliminacijska selekcija, koja se temelji na izbacivanju loših jedinki (onih sa manjim fitnesom). Za svaku jedinku određuje se njena kazna (eng. *penalty*) (razlika između maksimalnog i njenog fitnesa), te ta kazna predstavlja osnovu prema kojoj se vrši selekcija.

Sukcesivnim postupkom, iz generacije u generaciju, dobijaju se sve bolje i bolje jedinke (jedinke sa sve boljim i boljim fitnesom), te ovakav postupak po pravilu dovodi do pronalaženja optimalnog rješenja [20]

Osnovni motiv za upotrebu selekcije koja je zasnovana na muzičkim elementima populacije (eng. *music aware selection*) je činjenica da, u muzici, često nije cilj pronaći jedinku sa najboljim fitnessom (već je ranije rečeno da je računanje samog fitnessa problematično, ma kako ono bilo organizovano), već da se napravi prostor “dobrih” jedinki. Ovo zapravo znači da računanje fitnessa podliježe dodatnoj provjeri, te vrijednost fitnessa pojedinih jedinki nije isključiva ili presudna za odluku da li će jedinka preživjeti, ili u kojoj mjeri će učestvovati u daljoj reprodukciji. Ovo se postiže ili postavljanjem dodatnih kriterijuma koji ispituju podesnost jedinke u odnosu na druge jedinke ili se, što je čest slučaj, uključuje i ljudski faktor odlučivanja.

Odabir jedinki potpuno slučajnim izborom se vrši u slučajevima kada se jedinkama uopšte ne računa fitness. Primjeri upotrebe slučajne selekcije dati su u [21], [22].

Ukrštanje

Ukrštanje obezbjeđuje mehanizam za kombinovanje genetskog materijala dvije ili više jedinki. Ovim kombinovanjem, cilj je dobiti nove jedinke (potomke) koje sadrže kvalitetne dijelove (gene) svojih roditelja.

Standardno, koriste se tri vrste ukrštanja [16]: jednopoziciono (eng. *one-point crossover*), dvopoziciono (eng. *two-point crossover*) i uniformno ukrštanje (eng. *uniform crossover*). Izbor načina ukrštanja zavisi od problema koji se posmatra. Po osnovnom principu, u slučajevima sa velikom međuzavisnošću gena u genetskom kodu obično se koristi jednopoziciono ukrštanje, dok je, recimo, uniformno ukrštanje najčešće pogodno primijeniti kada je međuzavisnost gena u genetskom kodu mala.

Jednopoziciono ukrštanje. Kod jednopozicionog ukrštanja, na početku se bira koliko se parova jedinki ukršta. Najčešće, bira se određen broj najboljih jedinki (odnosno parova najboljih jedinki), ili čitava populacija (odnosno $n/2$ parova, gdje je n broj jedinki u populaciji). Zatim se za svaki par jedinki, (uz eventualno datom i vjerovatnoćom – nivoom ukrštanja), na slučajan način bira pozicija za ukrštanje i vrši razmjena odgovarajućih gena datih parova, poslije izabrane pozicije. Šematski prikaz jednopozicionog ukrštanja prikazan je u Tabeli 4. Pozicija za ukrštanje je 6.

Tabela 4 Jednopoziciono ukrštanje

Prije ukrštanja	Poslije ukrštanja
XXXXXX XXXXXXXXXXXX	XXXXXX YYYYYYYYYYYY
YYYYYY YYYYYYYYYYYY	YYYYYY XXXXXXXXXXXX

Teorijski gledano, jednopoziciono ukrštanje ima i neke bitne nedostatke. Na primjer, skoro nikada neće biti razmijenjen prvi element, a posljednji će biti razmijenjen u skoro svakom slučaju. To nije pogodno ukoliko su prvi i posljednji element u uzajamnoj vezi.

Dvopoziciono ukrštanje. Kao i kod jednopozicionog ukrštanja, na početku se, po odgovarajućem principu, bira koliko se parova ukršta. Zatim se za svaki par jedinki biraju dvije pozicije i vrši se odgovarajuća zamjena gena datih parova između datih pozicija. Šematski prikaz dvopozicionog ukrštanja prikazan je u Tabeli 5. Pozicije za ukrštanje su 4 i 7.

Tabela 5: Dvopoziciono ukrštanje

Prije ukrštanja	Poslije ukrštanja
XXXX XXX XXXXXXXXXX	XXXX YYY XXXXXXXXXX
YYYY YYY YYYYYYYYYY	YYYY XXX YYYYYYYYYY

Kod uniformnog ukrštanja se za svaki par jedinki koje se ukrštaju generiše maska na slučajan način. Maska je binarni niz iste dužine koliko ima i pojedinačnih elemenata (bitova) u svakoj jedinki. Ako je u generisanom nizu na i -tom elementu nula, tada se na i -tom mjestu vrši zamjena odgovarajućih elemenata (bitova) jedinki, a ako je vrijednost tog elementa jedinica, tada se zamjena ne vrši. Šematski prikaz uniformnog ukrštanja prikazan je u Tabeli 6.

Tabela 6 Uniformno ukrštanje

Maska 1101001100110001	
Prije ukrštanja	Poslije ukrštanja
XXXXXXXXXXXXXXXXXX	XXYXYXXYXXYYXX
YYYYYYYYYYYYYYYY	YYXYYXXYXXYYXX

Ako jedinka predstavlja niz od više tonova, izbor pozicije (ili pozicija) za ukrštanje jedinki može se (a vjerovatno u većini slučajeva i treba) dodatno ograničiti. Ovdje se prvenstveno misli na problem koji može nastati pokušajem razdvajanja jedinice na “pogrešnom” mjestu. Na primjer, pretpostavimo da se nota sastoji od niza od n bitova, (recimo 4 bita), a čitava kompozicija sadrži od m nota (recimo 20), pa se kompozicija predstavlja sa ukupno 80 bita. Tada bi ukrštanje za izbor pozicije 28 značilo da se prvih sedam nota prve kompozicije kombinuje sa posljednjih 13 nota druge. Sve pozicije, koje u ovom slučaju nisu djeljive sa 4 bi vršile podjelu na jednoj noti (a ne između neke dvije), a to (iako teoretski može) ne treba biti cilj. Pored ovoga, ukrštanje na “pogrešnoj” poziciji može dovesti i do neželjenih poremećaja u ritmu.

Mutacija

Mutacija sprečava pojavu tzv. prerane konvergencije, tj. kada se, uz pomoć drugih operatora (prije svega ukrštanja i selekcije) najbolje jedinice prevedu u neki oblik stacionarnog stanja, a rješenje koje je postignuto nije dovoljno dobro. Primjenom mutacije postiže se mogućnost izlaska iz takvog stanja, te dalje poboljšanje jedinki.

U opštem, najčešće korištenom načinu predstavljanja jedinki – binarnom kodiranju – mutacija u većini slučajeva podrazumijeva promjenu pojedinačnih bitova. Po standardnom sistemu, generiše se (može i više od jednog) slučajan indeks – pozicija (najčešće po normalnoj raspodjeli), pa se vrijednost bita na tom indeksu (tim indeksima) mijenja iz nule u jedinicu, ili iz jedinicu u nulu.

U slučaju binarnog kodiranja muzičke kompozicije, može se smatrati da je ovakva mutacija pogodna za primjenu. Sa druge strane, u literaturi se često sreću i drugi operatori mutacije, koje su specifične za reprezentaciju jedinki preko niza nota, gdje je svaka nota najmanji integrisani dio jedinice. U ovim situacijama, može se realizovati više različitih operatora mutacije, među kojima izdvajamo [18],[23]:

Zamjena jedne note: Podrazumijeva izbor jedne note iz čitave kompozicije i zamjenu ili sa novom generisanom notom ili, u drugoj varijanti, notom koja je u odnosu na datu, u nekom kraćem intervalu, ili se razlikuje od nje za čitavu oktavu.

Zamjena dvije susjedne note. Na slučajan način se bira indeks note i ta nota mijenja mjesto sa susjednom.

Zamjena dva segmenta. Na slučajan način se biraju dva segmenta kompozicije iste dužine (recimo istog broja nota), te se vrši zamjena datih segmenata

Transponovanje: Na slučajan način se bira segment kompozicije i vrši se njegovo transponovanje za odgovarajući broj (polu)tonova – najčešće za neki “skladan” interval, kao što je terca ili čista kvinta

Permutovanje: Na slučajan način se bira segment, te se note u tom segmentu permutuju

Sortiranje u rastućem poretku: Na slučajan način se bira segment (niz tonova) i taj segment (niz tonova) se sortira u rastućem poretku.

Sortiranje u opadajućem poretku: Na slučajan način se bira segment (niz tonova) i taj segment (niz tonova) se sortira u opadajućem poretku.

Preraspodjela trajanja: Na slučajan način se bira segment i permutuje se raspodjela trajanja tonova, dok se visina tonova ne mijenja.

Preraspodjela visina: Na slučajan način se bira segment i permutuje se raspodjela visina, dod se trajanje tonova ne mijenja.

Jednostavno kopiranje. Kopiranje jednog segmenta i njegovo “lijepljenje” preko drugih segmenata.

Kopiranje i mutacija: Kopiranje jednog segmenta, primjena neke od mutacija na taj segment i lijepljenje (izmijenjenog) segmenta preko drugih segmenata. Kao mutaciju, ima smisla primjenjivati transponovanje, sortiranje, preraspodjelu trajanja ili preraspodjelu visina.

Spajanje više istih tonova kraćeg trajanja u jedan ton, ekvivalentnog dužeg trajanja. Podrazumijeva spajanje više jednakih tonova u jedan, da bi se izbjeglo ponavljanje jednog tona, koje može biti uzrokovano mutacijama sortiranja ili ukrštanjem.

Obrtanje redoslijeda nota. Na slučajan način se bira jedan segment i taj segment se mijenja segmentom gdje je redoslijed nota obrnut u odnosu na početni.

Dodavanje note. Na slučajan način se bira pozicija na koju se dodaje nota, koja se može birati ili slučajno, ili iz nekog ograničenog skupa. Nezgodan operator zbog mogućeg poremećaja u ritmu.

Brisanje note. Na slučajan način se bira nota i briše se. Isto kao i prethodni operator, nezgodan za primjenu zbog mogućeg poremećaja u ritmu.

Zamjena

Kod opštih genetičkih algoritama, zamjena se uglavnom povezuje sa selekcijom, dodavanjem nove generacije jedinki staroj, računanjem fitnesa svake pojedinačne jedinke i odabirom odgovarajućeg broja jedinki³⁷. Većina sistema koristi i neki oblik elitizma. Kod genetičkih algoritama koji se koriste u komponovanju, uobičajena su dva načina zamjene: generacijski i kontinualni. Kod generacijske zamjene, na trenutnoj populaciji se vrši zamjena jedinki tek kada se formira kompletna nova generacija (odnosno nove jedinke koje su dobijene primjenom genetskih operatora). Kod kontinuirane zamjene, svaka nova jedinka (ili dvije po dvije), odmah nakon kreiranja ulazi u "proces" selekcije i eventualno zauzima mjesto u novoj generaciji populacije. Da bi se izbjeglo eventualno dupliranje jedinki u populaciji (što se u većini slučajeva izbjegava), pri zamjeni jedinki u populaciji se često ubacuje i dodatni sistem provjere.

³⁷ U većini genetičkih algoritama broj jedinki populacije je konstantan.

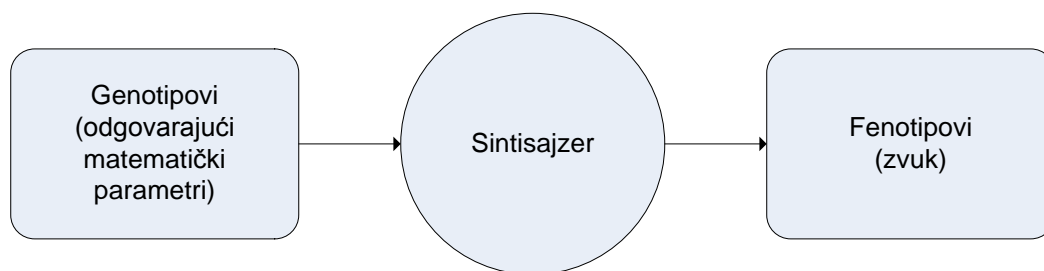
Genetički algoritmi i dizajniranje zvuka

Kao što je već rečeno, genetički algoritmi se u muzici najviše koriste za komponovanje muzike i dizajniranje zvuka. U ovom poglavlju se ukratko analizira upotreba genetičkih algoritama u dizajniranju zvuka.

Intenzivan razvoj i upotreba elektronske muzike³⁸ posljednjih nekoliko decenija prouzrokovali su i ubrzani razvoj tehnologija za dizajniranje zvuka. Razumijevanje čitavog procesa dizajniranje zvuka podrazumijeva dosta široko znanje iz drugih naučnih oblasti, prije svega fizike, elektrotehnike i elektronike (zvučni talasi, proizvodnja i upravljanje signalima, digitalna obrada signala i sinteza signala), te bi takva detaljnija analiza po obimu znatno izlazila iz okvira ovog rada. U ovom dijelu rada se polazi od matematičkog modela (načina) predstavljanja digitalnog zvučnog signala (pomoću odgovarajuće sinusoide), dok se detaljno ne objašnjavaju načini (i razlozi) kombinovanja različitih zvukova i njihova fizička interpretacija.

Jasno je da se kombinovanjem i modifikovanjem zvukova dobijaju novi zvukovi. Na čitavom prostoru tako dobijenih potencijalnih zvukova javlja se potreba za pretraživanjem i optimizacijom tog pretraživanja, u cilju pronalaska dovoljno dobrog rješenja (zvuka). Upravo tu genetički algoritmi pronalaze svoju primjenu.

Upotreba genetičkih algoritama u dizajniranju zvuka se odnosi na upravljanje parametrima koji definišu zvuk. Podrazumijeva se upotreba odgovarajuće tehnike sintetizovanja zvuka, ili se, na osnovu ulaznog signala mijenjaju parametri koji opisuju ulazni zvučni signal i tako dobija izlazni, drugačiji. Parametri koji utiču na sintezu zvuka konstituišu jedinice (genotipove) koji predstavljaju matematičke modele, odnosno algoritme koji proizvode zvuk, dok odgovarajući zvukovi predstavljaju fenotipove [24]. Na Slici 6 prikazana je veza između genotipova i fenotipova, tj. uzajamno mapiranje, koje se odvija preko posrednika – sintisajzera



Slika 6: Mapiranje genotipova i fenotipova

Postoje dvije glavne kategorije upotrebe genetičkih algoritama [25]:

³⁸ Elektronska muzika predstavlja dosta širok pojam. U ovom kontekstu, može se prihvatiti definicija elektronske muzike kao muzike koja uključuje elektronske muzičke instrumente i elektronsku tehnologiju njene proizvodnje. Prvenstveno se misli na muziku napravljenu ili putem računara ili putem muzičkih sintisajzera

1. Optimizacije tehnike za određivanje parametara koji dobro opisuju neki posmatrani zvuk,
2. Istraživanje prostora zvukova u potrazi za novim zvukovima.

Osnovna razlika između ova dva tipa algoritama je finte funkcija. Dok je u prvom slučaju fitness fenotipova dat stepenom sličnosti sa ciljnim zvukom, fitness u drugom slučaju zavisi od djelovanja korisnika na algoritam. Sistem zapravo sintetiše fenotipove (tj. zvukove), dok ih korisnik rangira prema sopstvenoj procjeni.

Pod određivanjem parametara koji opisuju neki posmatrani zvuk podrazumijeva se razvoj i primjena odgovarajućeg algoritma u okviru koga se, kombinovanjem parametara, želi dobiti što bolja aproksimacija željenog zvuka. Kreiranje ovakvih algoritama je težak problem i obično zahtijeva pažljivu analizu željenih karakteristika završnog zvuka, ali i teorijskih ograničenja samog algoritma.

Primjer optimizacije procesa kreiranja zvuka

Kao dobar primjer genetičkog algoritma za sintezu zvuka u literaturi često se navodi [26]. Ovdje ukratko prenosimo ideju, način realizacije i rezultat.

U [26] je predstavljen genetički algoritam koji pretražuje prostor svih algoritama za sintezu zvuka i primjenom odgovarajućih genetičkih operatora pronalazi one jedinke (algoritme) koji sintetišu “bolji” zvuk. Smatra se da svaki pojedinačni algoritam generiše jedan zvuk. Algoritmi se opisuju na dvojak način: kao topološki graf, odnosno necikličan graf (stablo) sa usmjerenim granama. Genetički algoritam koristi stabla da bi se od njih razvio odgovarajući topološki objekat na osnovu kog se renderuje³⁹ zvuk. Dobijeni zvukovi se rangiraju na osnovu sličnosti sa ciljnim zvukom. Sličnost se ispituje na osnovu fizičkih osobina zvuka koje se mjere ili prikazuju pomoću odgovarajućih elektronskih uređaja. Ovaj pristup omogućava sugerisanje novih algoritama za sintezu zvuka i taj postupak se nastavlja sve dok se ne postigne tražena sličnost sa ciljnim zvukom.

Kreiranje svakog pojedinačnog zvuka zasnovano je na primjeni niza funkcionalnih blokova. Međusobno povezani funkcionalni blokovi čine topologiju. Prostor svih algoritama, tj. odgovarajućih topologija čine topološki prostor: prostor pokriven svim mogućim kombinacijama funkcionalnih blokova i njihovih veza. Ovaj prostor se može suziti tako što se ograniči broj tipova blokova, kao i ukupan broj blokova koji učestvuju u gradnji jedne topologije. Pored toga, posmatraju se samo “validne” topologije, tj. one kod kojih su pravilno spojeni ulazni i izlazni elementi signala. Upotreba tako dobro složenih funkcionalnih blokova čini mrežu međusobno povezanih jedinica – generatora koja predstavlja arhitektura sintisajzera zvuka.

Sugerisana su četiri tipa blokova: *source*, *render*⁴⁰, Tip A, Tip B. Tip *source* predstavlja početni blok koji ima samo jedan izlaz. Tip TipA ima dva ulazna a jedan izlazni podatak, tip Tip B ima jedan ulazni, a dva izlazna, dok je tip *render* jedan ulazni podatak, a nema izlazne podatke. U Tabeli 7 su prikazani vrste funkcionalnih blokova.

³⁹ renderovanje - proces kreiranja zvuka na osnovu nekog modela

⁴⁰ nema smisla prevoditi na naš jezik

Tabela 7: Funkcionalni blokovi koji kreiraju zvuk

Naziv	Tip	Funkcija
Oscilator	Tip B	sinus, triangularna ⁴¹ funkcija, kvadratna funkcija
Filter	Tip B	Daje dvije rezonantne vrijednosti
Split	Tip B	Podjela na dvije vrijednosti
Constant	Tip B	Konstantna funkcija
Add	Tip A	Sabiranje (dva u jedan)
Mult	Tip A	Množenje (dva u jedan)
Filter+Oscilator	Tip A	Oscilator sa filtriranim izlazom (dva u jedan)
Source	Source	Jedan izlaz
Render	Render	Jedan ulaz

Direktno pretraživanje ovakvog topološkog prostora se pokazalo teškim i neefikasnim. Pretraživanje može podrazumijevati dodavanje, brisanje ili pomjeranje funkcionalnih blokova, što se u topološkim grafovima koji su često ciklični ovaj proces dosta komplikuje. Na primjer, ako je dodat novi blok, moguće je da je promijenjen odnos ulaznih odnosno izlaznih rezultata. Na takvoj "lošoj" topologiji moraju se primijeniti dodatna dodavanja ili brisanja da bi ona postala validna, što često previše opterećuje samu pretragu, posebno ako je topologija jako velika. Da bi se izbjeglo pretraživanje po topološkom grafu, uvodi se odgovarajući ekvivalent: usmjereno (neciklično) stablo. Čvorovi drveća imaju instrukcije koje utiču na razvoj odgovarajuće topologije (embriona). Stabla zapravo opisuju (kodiraju) kako topologija „raste“. Rezultat nakon primjene instrukcija u stablu na topologiju (embrion) je potpuno razvijena i validna topologija. Prednost ovakvog načina reprezentovanja je lakše manipulisanje čistom grafovskom strukturom (stablom) u odnosu na složeniju, topologiju. Na primjer, lakše je kopirati, mutirati ili ukrštati stabla nego topologiju. Prema [27] postoje tri vrste instrukcija koje predstavljaju čvorove stabla:

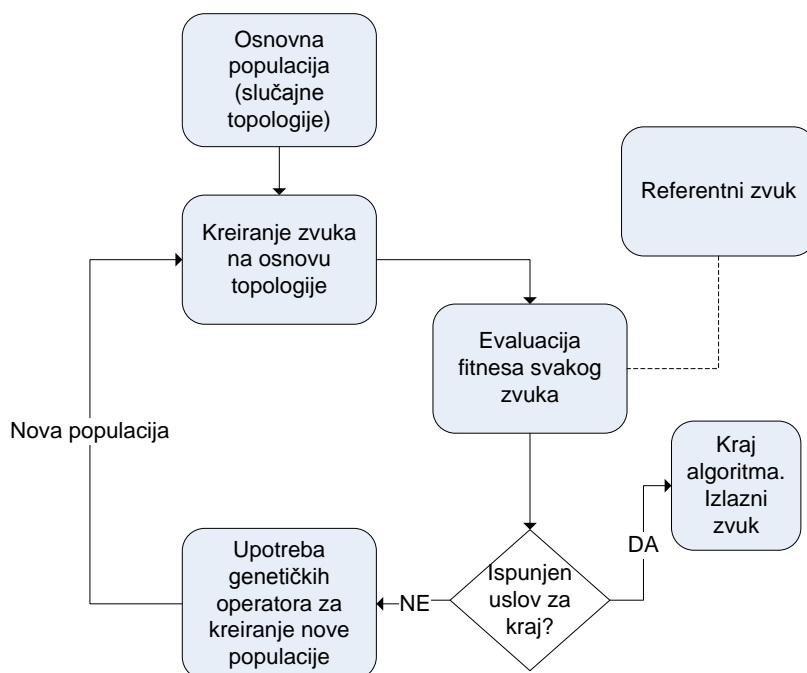
1. Funkcije koje modifikuju topologiju (eng. *Topology Modifying Functions*). Izvršavaju niz operacija jednu za drugom ili paralelno na izabrani blok. Ove operacije ubacuju nove blokove i vrše dodatne operacije da bi topologija ostala validna.
2. Funkcije koje kontrolišu razvoj (eng. *Development Control Functions*). Ne mijenjaju topologiju, ali ubacuju „nop“ ili „end“ operacije koje odlažu ili zaustavljaju granjanje u razvoju topologije
3. Funkcije koje konstruišu podstabla (eng. *Construction Continuing Subtrees*). Biraju jednu od tri mogućnosti: označavaju tip trenutnog bloka ili preduzimaju jednu od druge dvije funkcije na dati blok.

Uvođenjem odgovarajućih stabala za svaku topologiju, problem pretraživanja prostora zvukova (tj. topologija) se svodi na jednostavnije pretraživanje stabala, u cilju pronalazjenja onih koji opisuju topologiju koja nosi informaciju o zvuku bliskom ciljanom. Svako stablo se preslikava (odgovara) samo jednoj topologiji, što u obrnutom smjeru nije slučaj: jedna topologija može biti predstavljena sa više stabala. Pretpostavka je da je da se stablima ipak pokriva dovoljno topologija, tako da se među njima može naći dovoljno dobro rješenje.

⁴¹ poznata i kao „šesir“ funkcija $f(t) = \max\{0, 1 - |t|\}$

Opis jednog algoritma za optimizovanje procesa kreiranja zvuka

Opisana stabla čine populaciju u genetičkom algoritmu. Svakom stablu se pripisuje odgovarajuća fitness funkcija. Nova populacija se kreira korištenjem genetičkih operacija: kopiranje, ukrštanje i mutacija. Vršni se tzv. rulet selekcija (jedinke sa boljim fitnessom imaju veću šansu da prežive). Fitness se računa na osnovu pojedinačnih karakteristika zasnovanih na nekom testu ili poređenju sa ciljnim zvukom. Proces generisanja novih jedinki se ponavlja sve dok se ne postigne odgovarajući fitness, ili dok se ne dostigne maksimalan broj iteracija. Prema [27], ovakav sistem dovodi do zadovoljavajućeg rješenja. Određivanje fitnessa posmatrane jedinke je proces koji uključuje dva dijela: na osnovu generisane jedinke (stabla) formira se odgovarajuća topologija, te se na osnovu topologije generiše zvuk koji se dalje poredi na sa ciljnim zvukom. Šema algoritma prikazana je na Slici 7.



Slika 7: Šema genetičkog algoritma za optimizovanje procesa kreiranja zvuka

U ovom rješenju fitness funkcija je zasnovana na određivanju sličnosti dobijenog zvuka sa ciljnim. U literaturi su navedeni različiti kriterijumi za mjerenje sličnosti. Propisuju se dva glavna načina određivanja sličnosti: analitički i perceptualni. Prema [26], poređenje zvukova može biti zasnovano na određivanju težine klastera⁴² (eng. *cluster weighted*) i određivanja udaljenosti između odgovarajućih klastera dva posmatrana zvuka. Perceptualna tehnika podrazumijeva procjenu sličnosti na osnovu samih “zvučnih” osobina, kao što je, na primjer, poređenje odgovarajućih spektrograma. U razmatranom radu kombinovane su obje metode, prvo se na osnovu analitičke metode sužavao skup svih generisanih zvukova, a nakon toga se odabrani zvukovi daju na perceptualnu procjenu. Primjeri generisanih tonova mogu se preuzeti na sajtu autora⁴³.

⁴² Klaster možemo shvatiti kao jedan podrazdio čitavog zvučnog signala

⁴³ <http://www.ragomusic.com/research/ml/>

Genetički algoritmi u komponovanju muzike – sopstveni primjer

Cilj i ideja algoritma

Kao ilustraciju primjene genetičkih algoritama u komponovanju muzike navodimo algoritam koji, po mišljenju autora, daje kompozicije zadovoljavajućeg kvaliteta. Prije detaljnog opisa, analize algoritma i dobijenih rezultata, ukratko navodimo cilj i ideju samog algoritma:

1. Algoritam ima za cilj komponovanje kratkih kompozicija (na primjer četiri četvoročetvrtinska takta).
2. Kompozicije se predstavljaju nizom (brojeva) koji nosi podatke o tonovima i njihovom trajanju.
3. Opšti ulazni parametri određuju: dužinu kompozicije, tonalitet, broj i opseg dozvoljenih tonova, broj iteracija, kriterijum za završetak algoritma, način interpretacije rezultata algoritma i slično.
4. Ulazni parametri koji utiču na procjenu kvaliteta kompozicije određuju: vrijednosti koje upućuju na sličnost kompozicije sa referentnom kompozicijom (ili referentnim vrijednostima), vrijednosti intervala, skup „dobrih“ i „loših“⁴⁴ tonova, dozvoljeno odstupanje (varijancu) od propisanih referentnih vrijednosti, kao i težinske faktore koji utiču na važnost pojedinačnih kriterijuma.
5. Važan dio algoritma je uspostavljanje kriterijuma koji određuju kvalitet kompozicije. Ovi kriterijumi se prvenstveno odnose na ocjenu intervala između uzastopnih tonova i odstupanje od referentnih vrijednosti.
6. Prostor kompozicija se pretražuje po principu genetičkih algoritama, u cilju pronalaska dovoljno dobre kompozicije⁴⁵. Polazi se od skupa slučajno generisanih jedinki (kompozicija). Ovaj proces generisanja slučajnih kompozicija je djelimično kontrolisan ulaznim parametrima. Primjenom operatora genetičkih algoritama se, iz iteracije u iteraciju, teži dobijanju jedinke koja svojim kvalitetom zadovoljava uslov za prekid algoritma. U svakoj iteraciji se računa fitnes svih jedinki populacije, kreiraju se nove jedinke koje nastaju mutacijom trenutno najboljih jedinki, a potom i selekcija novih najboljih jedinki iz trenutne populacije.
7. Izlazni podatak iz algoritma je kompozicija, koja se, u zavisnosti od podešenih parametara i odvijanja iteracija smatra optimalnom.

Po izboru autora, algoritam je implementiran u programskom jeziku Java. Izlazni proizvod algoritma je odgovarajući muzički zapis na osnovu koga se može proizvesti neki od standardnih muzičkih izlaza. Za integrisano razvojno okruženje za algoritme realizovane u Javi korišteno je razvojno okruženje JCreator⁴⁶. U domenu koliko je korišteno, ovo javino okruženje je besplatno i javno dostupno. Kao muzički interfejs (za produkciju zvučnog zapisa ili kreiranje izlaznih MIDI fajlova) za sve primjere korišteno je rješenje JFugue⁴⁷. Za kreiranje notnog zapisa korišten je program Notation Musician⁴⁸.

⁴⁴ U nastavku će detaljnije biti objašnjen motiv za proglašenje „dobrih“ i „loših“ tonova.

⁴⁵ Kvalitet kompozicije zavisi od zadatih parametara

⁴⁶ www.jcreator.com

⁴⁷ www.jfugue.org

⁴⁸ www.notation.com

Softverski paket *JFugue*

JFugue je *open source*⁴⁹ Java API⁵⁰ za programiranje muzike. Kao osnovna funkcionalnost ovog rješenja prepoznaje se mogućnost direktne reprodukcije zvuka na osnovu tekstualnog zapisa odgovarajućih tonova. U sintaksi, najjednostavniji način proizvodnje zvuka bi izgledao ovako:

```
Player player = new Player();  
player.play("C D E F G A B");
```

JFugue omogućava predstavljanje nota, akorada, instrumenata i čitavih kompozicija uz pomoć odgovarajućih „muzičkih stringova“. Muzika se može proizvoditi u realnom vremenu, ili snimati u MIDI zapis. Takođe, omogućen je sistem preuzimanja muzike iz postojećih MIDI zapisa. Omogućeno je efikasno i funkcionalno upravljanje dijelovima kompozicije, kao što je promjena oktava, skala, trajanja tonova, zamjena nota, instrumenata, akorada i slično.

Kako se navodi u specifikaciji, JFugue je pogodan za:

- evolucionarnu i uopšte algoritamsku muziku;
- muzičke editore, ritam ili bubanj mašine;
- džez improvizaciju, imitiranje kompozitora i vještačku inteligenciju u muzici;
- proceduralnu sintezu i virtuelne instrumente;
- dinamičko podešavanje muzike, adaptivnu muziku (za primjenu u video igricama), igrama koje zahtijevaju muzičko znanje i slično;
- eksperimentisanje u teoriji muzike i komponovanju.

U osnovi, JFugue se oslanja na javine sposobnosti da pomoću MIDI protokola proizvodi muziku. Java MIDI koristi *Java Sound engine*, koji opet koristi banku odgovarajućih zvukova da bi generisao zvukove pomoću sintisajzera.

Neki primjeri upotrebe JFugue elemenata koji su korišteni u radu

Klasa Player

Klasa Player priprema odgovarajući muzički zapis (u javinoj sintaksi) za neku od odgovarajućih interpretacija (najčešće direktno u zvuk ili u MIDI zapis), te tako predstavlja posrednika između samog javinog koda u programu i odgovarajućih javinih sintisajzera i MIDI kontrola.

Najvažniji upotrebnici su oni koji omogućavaju sviranje (nekoliko preopterećenih metoda `play(...)`⁵¹, te metodi koji omogućavaju učitavanje (`loadMidi(...)`) i snimanje (`saveMidi(...)`) MIDI zapisa.

Primjer upotrebe klase Player [28]:

⁴⁹ Ustaljen naziv i u našem jeziku, vezuje se za softverska rješenja „otvorenog koda“, sistema preuzimanja rješenja u izvornom obliku, pogodnom za modifikaciju, prilagođavanje i integraciju u druge sisteme

⁵⁰ Application Programming Interface (API) – skup različitih softverskih elemenata (funkcija, struktura, objekata, protokola...) koji djelimično ili potpuno omogućava postizanje funkcionalnosti razvijane aplikacije

⁵¹ U sintaksi nazivMetoda(...) tačkice unutar zagrade označavaju da dati metod uzima argumente, koji se u objašnjenju samog metoda ne navode.

```
Player player = new Player();
player.play("C");
player.play("C7h");
player.play("C5maj7w");
player.play("G5h+B5h+C6q_D6q");
player.play("G5q G5q F5q E5q D5h");
player.play("T[Allegro] V0 IO G6q A5q V1 A5q G6q");
player.play("V0 Cmajw V1 I[Flute] G4q E4q C4q E4q");
player.play("T120 V0 I[Piano] G5q G5q V9 [Hand_Clap]q Rq");
```

Klasa Pattern

Objektima klase Pattern predstavljaju se muzički segmenti. Predstavljanjem segmenata kao paterna, korisniku se pruža mogućnost kombinovanja i modifikovanja dijelova kompozicija na različite načine, kao što je kopiranje ili dodavanje novih dijelova, promjena oktave, tonaliteta, instrumenta, tempa i slično. Objekti klase Pattern nose kompletan set informacija o jednom dijelu kompozicije.

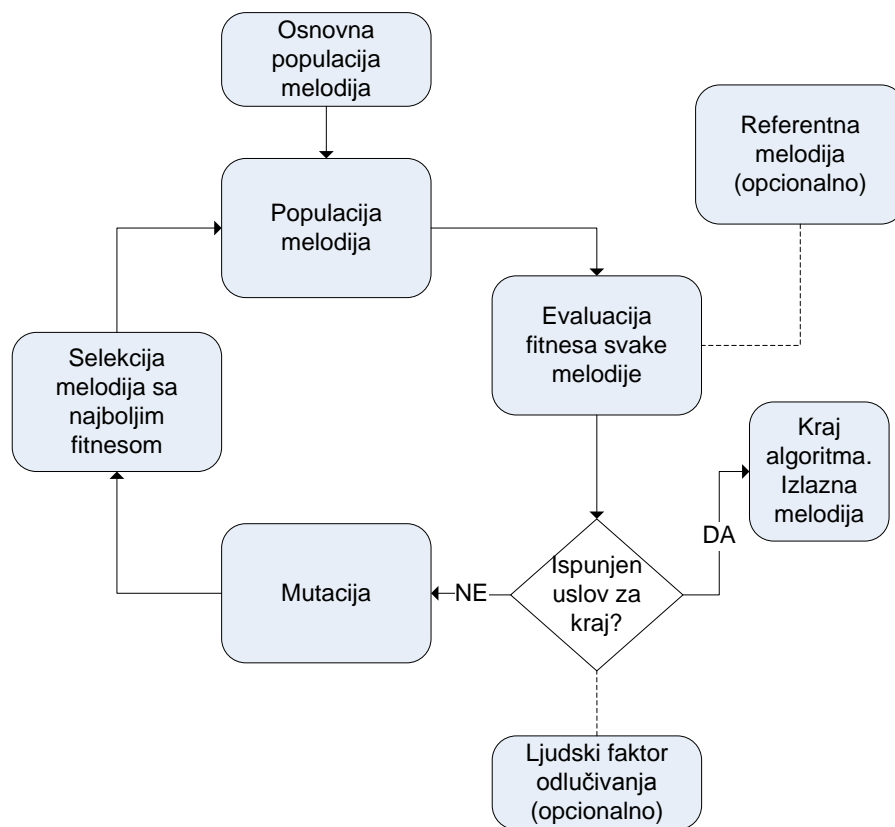
Najvažniji metodi klase Pattern se odnose na dodavanje novih segmenata na postojeći (nekoliko preopterećenih metoda `add(...)`), označavanje dijelova paterna koji se ponavljaju, snimanje ili učitavanje paterna u/iz tekstualnog fajla i slično.

Detaljan opis algoritma – opšti principi funkcionisanja

U algoritmu se formira populacija jedinki koje unaprijed imaju ritam sličan referentnoj jedinki (raspored otkucaja kod svake jedinke je potpuno isti kao kod referentne, a eventualni „poremećaji“ u ritmu mogu nastati usljed generisanja pauza na različitim mjestima). Svaka jedinka može da predstavlja kompletnu kompoziciju, ili jedan njen dio. Fitnes funkcija se računa na osnovu tri kriterijuma, gdje je svakom kriterijumu moguće dodijeliti odgovarajuću „težinu“, tj. definisati uticaj kriterijuma na čitav fitnes. Populacija sadrži unaprijed definisan broj jedinki koje su sortirane po fitnessu. U svakom koraku iteracije, na jednoj trećini jedinki (uvijek onih sa najboljim fitnessom) se primjenjuje jedan od moguća tri operatora mutacije. Vršiti se generacijski odabir nove populacije (od svih ukupno generisanih novih jedinki, zajedno sa svim starima bira se nova generacija za sljedeću iteraciju). Operator ukrštanja se ne koristi. Algoritam prestaje sa radom ili kada se dostigne maksimalan broj iteracija, ili kada se formira jedinka sa dovoljno dobrim fitnessom. Kvalitet jedinke je obrnut u odnosu na veličinu fitnesa (zapravo se umjesto fitnesa koristi suprotan termin - funkcija kazne (eng. *penalty factor*). Jedinka je sve „bolja“ kako joj se smanjuje fitnes.

Uvođenje referentne jedinke je opciono, a može se smatrati korisnim i praktičnim ukoliko želimo da naša kompozicija ima neki prepoznatljiv ritam (raspored trajanja nota i pauza), ili, što je češći slučaj, ako želimo da odabir trajanja nota ne prepustimo slučajnom generatoru (mnogo je veća vjerovatnoća da će se slučajnim izborom generisati prilično nepravilan i neugodan ritam). Pored determinisanja ritma, referentna jedinka utiče i na fitnes, ali samo u mjeri unaprijed izračunatih vrijednosti, koje se u algoritam mogu unijeti proizvoljno i nezavisno od referentne jedinke.

Na Slici 8 su prikazani glavni elementi algoritma. Na osnovu početnih parametara generiše se početna populacija od ukupno n jedinki. Nakon toga počinje iterativni proces. U svakoj iteraciji se za svaku jedinku trenutne populacije računa fitness. Nakon toga se lista jedinki populacije sortira po fitnessu. Na osnovu fitnessa najbolje jedinke ispituje se da li je zadovoljen uslov za kraj algoritma. Ako jeste, algoritam prekida sa radom i odgovarajuća najbolja jedinka (kompozicija) se proglašava rezultatom izvršenja algoritma. Ako nije, ulazi se u proces kreiranja novih jedinki. Od svih jedinki trenutne populacije biraju se (konkretno, trećina od ukupnog broja) najbolje jedinke i na njima se izvršavaju operatori mutacije, te se tako dobijaju nove jedinke. Svaka nova jedinka se dodaje u listu jedinki. Nakon primjene svih mutacija na izabrane jedinke, lista jedinki se ponovo sortira (po fitnessu). Nakon toga se iz liste izbacuju prvo duplikati (jedinke sa istim fitnessom), a potom i „višak“ jedinki, tako da na kraju u listi ponovo ostane tačno n jedinki. Iterativni postupak se ponavlja sve dok ne bude ispunjen kriterijum za prekid, ili se ne dostigne maksimalan broj iteracija.



Slika 8: Šema funkcionisana genetičkog algoritma korištenog za komponovanje muzike

Uključivanje ljudskog faktora je opcionalno i u osnovnoj realizaciji algoritma nije posmatrano.

Kreiranje jedinke

U početku se bira referentna jedinka, na osnovu koje se određuju opšti parametri: mjera, tonalitet, broj i spisak dozvoljenih (polu)tonova, ukupno trajanje jedinke (u otkucajima ili broj taktova), najkraća dužina

(najveći zajednički djelilac svih trajanja), broj najkraćih dužina u jednom taktu, kao i raspored otkucaja u jedinki. Neka se čitava kompozicija sastoji od m taktova, a svaki takt od po p otkucaja. Neka u jednom otkucaju ima q jediničnih dužina. Tada se svaka jedinka sastoji od ukupno $n = pmq$ najkraćih dužina.

Jedinka se, dakle, predstavlja nizom dužine n , gdje je svaki element niza iz intervala $[0, bdt+1]$, gdje je bdt ukupan broj dozvoljenih tonova. Nula je rezervisana za pauzu, a posljednji broj ($bdt+1$) za oznaku da je prethodni ton (ili pauza) produžen za jednu najkraću dužinu.

Svaka jedinka se generiše na proizvoljan način, uz dva ograničenja:

- svi elementi su iz intervala $[0, bdt+1]$,
- i – ti element je jednak upravo $bdt+1$ (označava produženje prethodnog tona) akko je odgovarajući (za to i) element niza referentne jedinice jednak $bdt+1$. Time se očuvava ritam.

Određivanje fitnessa jedinice

Pošto imamo referentnu jedinku (odnosno referentne vrijednosti), određivanje fitnessa je (ne u potpunosti) povezano sa procjenom koliko naša jedinka „liči“ na referentne. Pored toga, s obzirom na to da su dozvoljeni svi polutonovi iz posmatranog intervala dozvoljenih tonova, moguće je da se prilikom generisanja jedinice dobiju intervali koji su cijenjivi - skladni, ali ne pripadaju traženom tonalitetu. Stoga je potrebno da na završnu vrijednost fitnessa utiče i broj tonova van tonaliteta. Kvalitet jedinice je obrnuto proporcionalan ovom broju. Znači, tonovi van datog tonaliteta su dozvoljeni, ali je jedinka ipak bolja što ih ima manje. Ovakav pristup se poklapa sa ranije navedenim „mjerljivim“ elementima kompozicije.

Sličnost jedinice sa referentnom jedinkom se određuje na osnovu definisane „udaljenosti“ jedinice od referentne jedinice. Računanje udaljenosti se vrši takt po takt. Grubo govoreći, udaljenost između jedinice, odnosno odgovarajućih taktova je zasnovana na broju i tipu „dobrih⁵²“ intervala, kao i njihovog rasporeda po taktovima. U slučaju da se referentna jedinka ne koristi, tada se parametri koji utiču na poređenje moraju „ručno“ definisati. Matematički gledano, sličnost je zasnovana na određivanju aritmetičke sredine vrijednosti intervala u taktu i odgovarajuće varijance obje kompozicije, za svaki takt pojedinačno. Potom se posmatraju razlike između odgovarajućih vrijednosti, koje se nakon toga sabiraju (uz eventualno množenje nekim težinskim faktorima). Proces određivanja je sljedeći:

Određivanje brojne vrijednosti svake pojedinačne note. Prema načinu predstavljanja kompozicije, svakoj noti odgovara odgovarajući broj, tj. redni broj u odnosu na početnu notu.

Određivanje i ocjenjivanje svih intervala (u taktu). Interval čine dvije uzastopne note (pauze se preskaču). Ako posmatramo odgovarajući niz, svi intervali su vrijednosti razlika svaka dva uzastopna elementa koji su različiti od nule i od ukupne dužine (koji ne označava notu već produžetak trajanja). Tako, u odnosu na ukupan broj nota, intervala ima za jedan manje, i to u prvom taktu. Intervali koji su „na granici“ dva takta se vežu za drugu notu intervala, tj. za drugi od ta dva takta. Ocjenjivanje intervala

⁵² U najčešćem slučaju, dobri intervali su zapravo skladni, konsonantni intervali, ali se u opštem slučaju, za dobar interval može proglasiti bilo koji.

se po pravilu (opet ne obavezno) vrši tako što se „skladnijim“ intervalima daju veće vrijednosti. U tom slučaju, generalno, cjenjenije jedinke će biti one koje imaju više skladnih intervala⁵³, te ako je to cilj, to se i dobija. U Tabeli 8 su data dva prijedloga ocjenjivanja intervala. Treba napomenuti da je, zbog funkcionisanja algoritma (računanja fitnes funkcije), manja vrijednost intervala zapravo govori da se taj interval više cijeni. Primjeri vrednovanja intervala dati su u posljednje dvije kolone Tabele 8.

Tabela 8: Prijedlozi za vrijednost intervala

Intervalne kategorije	Intervali	Vrijednosti (prijedlozi)	
		I prijedlog	II prijedlog
savršeni konsonanti	prima, čiste kvarte i kvinte, oktava	1	1
nesavršeni konsonanti	mala i velika terca, mala i velika seksta	2	3
sekunde	mala i velika sekunda	3	1
septime	mala i velika septima	3	3
intervali veći od jedne oktave	svi intervali preko jedne oktave	5	5

Određivanje aritmetičke sredine i varijance. Aritmetička sredina i varijanca se računaju za svaki pojedinačni takt. Aritmetička sredina predstavlja srednju vrijednost svih vrijednosti intervala koji su prisutni u taktu.

$$a = \frac{1}{n} \sum_{i=1}^n x_i$$

gdje je x_i vrijednost i tog intervala, a n je broj intervala. Tako recimo, prema podacima iz Tabele 8, ako su svi intervali u taktu savršeni konsonanti, tada je aritmetička sredina jednaka jedinici.

Varijanca se računa kao srednja vrijednost sume kvadrata svih odstupanja intervala u od aritmetičke sredine, ili po formuli:

$$\sigma^2 = \sum_{i=0}^n (x_i - a)^2$$

Iz ove formule vidimo da je varijanca veća što je u taktu više (po skladnosti) „različitih“ intervala.

Znači, za svaki takt referentne i posmatrane jedinke se određuju ove vrijednosti. Informacija koja govori o sličnosti između dvije jedinke se dobija na osnovu formula,

$$f_1 = \sum_{i=1}^m \zeta_i (\mu_i - a_i)$$

⁵³ Na osnovu različitih eksperimenata sprovedenih u radu, pokazalo se da „optimizacija“ prema neskladnim intervalima (posebno prema velikoj ili maloj sekundi) može da da subjektivno lijepe dionice kompozicije, kao što su uzastopne skale više tonova, ili variranje oko jednog tona. Davanje veće vrijednosti septimama ili prekomjernoj kvarti, generalno ne daje subjektivno „lijepe“ kompozicije.

$$f_2 = \sum_{i=1}^m \eta_i (\sigma_i^2 - b_i^2)$$

gdje je ζ_i težinski faktor (značaj) razlike aritmetičkih sredina u i tom taktu, μ_i aritmetička sredina i -tog takta referentne melodije (ili unaprijed zadata vrijednost, ako je referentna jedinka ne koristi), a_i aritmetička sredina i -tog takta posmatrane jedinice, η_i težinski faktor (značaj) i te devijacije, σ_i^2 varijanca i -tog takta referentne jedinice (opet, ako se ne koristi, tada se unaprijed zadaje vrijednost), a b_i^2 varijanca i -tog takta posmatrane jedinice. Broj m predstavlja broj taktova. Ostavljanjem mogućnosti podešavanja vrijednosti za vrijednosti ζ_i i η_i za svaki takt pojedinačno, daje se mogućnost „balansiranja“ rasporeda intervala. Na primjer, na početku i na kraju kompozicije se ovim brojevima mogu dati veće vrijednosti, a u sredini manje, što znači da se na početku i na kraju više potencira sličnost sa referentnom jedinkom (ili unaprijed zadanim vrijednostima). U konkretnom primjeru, sve vrijednosti težinskih faktora su jednake jedinici.

Ukupna sličnost se definiše kao $f = \alpha f_1 + \beta f_2$.

α i β su globalni težinski faktori. U konkretnom primjeru oba globalna faktora su jednaka jedinici.

Jasno je, da ako su referentna i posmatrana jedinka jednake, vrijednost f će biti jednaka nuli. Obrnuto ne važi, vrijednost f može biti jednaka nuli i ako jedinice nisu jednake. To i opravdava „opcionalnost“ upotrebe referentne jedinice. Ono što ta informacija govori, jeste da tada posmatrane jedinice imaju sličan (ili isti) raspored intervala koji se jednako cijene, po taktovima.

Dalje, u algoritmu se posmatra i dodatni faktor koji utiče na fitnes – broj tonova koji su van propisanog tonaliteta. U principu, realizovan je opštiji slučaj, gdje se pravi lista „loših“ tonova⁵⁴. Računa se ukupan broj loših tonova u jedinki. Pauze se ignorišu (smatraju se „dobrim“ tonovima). Tako dobijamo vrijednost

$$g = \gamma \frac{1}{bl}$$

gdje je bl broj „loših“ tonova. γ je težinski faktor (značaj uticaja), a u konkretnom primjeru je jednaka jedinici.

Dakle, ukupan fitnes jedinice računa se po formuli

$$UkupanFitnes = f + g = \alpha \sum_{i=1}^m \zeta_i (\mu_i - a_i) + \beta \sum_{i=1}^m \eta_i (\sigma_i^2 - b_i^2) + \gamma \frac{1}{bl}$$

⁵⁴ U konkretnom rješenju, „loši tonovi“ su oni koji su van datog tonaliteta. U opštem slučaju, to može biti bilo koji ton.

Formiranje i organizacija populacije

Populaciju čine jedinke. Broj jedinki u populaciji predstavlja veličinu populacije. Ovaj broj može biti promjenljiv. Eksperimenti su pokazali da je za postizanje zadovoljavajućeg rješenja dovoljno nekoliko desetina (recimo do 50) jedinki. Struktura u kojoj se čuvaju jedinke (u konkretnim rješenjima redovno je riječ ili o nizovima ili o listama) je uređena po fitnessu. Inicijalna populacija podrazumijeva kreiranje slučajnih jedinki koje se smještaju u datu strukturu, koja se na kraju sortira. Proces sortiranja se poziva svaki put kada se završi generisanje novih jedinki (pomoću genetskih operatora).

Genetski operatori korišteni u algoritmu

Od raspoloživih genetskih operatora u algoritmu se koriste tri vrste mutacije i selekcija, dok je ukrštanje izostavljeno. Osnovni razlozi za izostanak operatora ukrštanja su:

- algoritmom se generišu relativno kratke jedinke⁵⁵, te nema velikog smisla ukrštati tako kratke kompozicije;
- primjenom tri vrste mutacije i dobrim balansiranjem parametara koji utiču na fitness se dostiže dovoljan kvalitet rezultata (ne uvijek, ali je čest slučaj da fitness dobijene jedinke bude jednak nuli), te ukrštanje (ili bilo koji drugi operator) ne može dalje optimizovati optimalno rješenje;
- Dobijene najbolje jedinke predstavljaju dobre „semplove“ za kreiranje novih većih (dužih) kompozicija, te bi nadogradnja ovog algoritma trebala ići u smjeru ukrštanja čitavih jedinki unutar tih dužih kompozicija, a to već, po zamisli autora, izlazi iz obima ovog rada;
- Pošto nije cilj razviti brz algoritam, već onaj koji u dozvoljenom vremenu uspijeva prepoznati dovoljno kvalitetnu jedinku, za svaku generaciju je ostavljena mogućnost generisanja ogromnog broja jedinki, od kojih se jako veliki broj njih selekcijom odbacuje. Na ovaj način se poštuje princip eksploatacije skupa svih jedinki, prije nego optimizovanja.

Mutacije

Po uzoru na korištenu literaturu, implementirane su tri različite mutacije. Predviđeno je da se za svaku mutaciju određuje da li će se i sa kojom vjerovatnoćom (u odnosu na druge mutacije) desiti. Dalje, postoji mogućnost izbora na koje jedinke (i koliko puta) će biti primijenjena mutacija. S obzirom na to da nema operatora ukrštanja, izabran je sistem da se mutacije primjenjuju na bolje jedinke i to više njih po jednoj jedinki. Na taj način se od dobrih jedinki pokušavaju napraviti još bolje. Sa druge strane, vrlo je moguće je da se primjenom mutacije fitness uopšte ne mijenja (iako se jedinka mijenja), tako da je moguća pojava više različitih jedinki sa istim fitnessom. Ovaj problem se rješava odgovarajućom selekcijom.

Mutacija 1: Zamjena tona za oktavu. Ova mutacija potencijalno smanjuje broj „velikih intervala“, tj. onih koji su veći od jedne oktave, koji se po standardnim postavkama algoritma ne cijene.

Mutacija 2: Promjena jednog tona. Ova mutacija omogućava „popravljanje“ fitnessa stare jedinke, u slučaju kada se mijenja ton koji sa svojim susjedima nije u „skladnom“ odnosu. U tom slučaju, zamjenom tona nekim drugim javlja se šansa da se fitness popravi. Po funkcionisanju selekcije, „kvarenje“ fitnessa

⁵⁵ Po izboru autora, generiše se četiri takta kompozicije. Algoritam pruža mogućnost generisanja proizvoljnog broja taktova.

(dobijanje lošijeg fitnesa u novoj jedinki) ne utiče na opšti kvalitet, jer će tom slučaju preživjeti stara jedinka.

Mutacija 3: Zamjena dvije uzastopne note. Na slučajan način se bira indeks na kome se nalazi nota, te se ona mijenja sa susjednom notom. Ovom mutacijom je moguće popraviti fitnes tako što se „okolni intervali“ mijenjaju i eventualno popravljaju.

Selekcija

Selekcija igra vrlo značajnu ulogu, s obzirom da se u svakoj iteraciji generiše veliki broj novih jedinki. Koristi se eliminacijska selekcija (jedinke koje su slabijeg fitnesa se izbacuju), uz dodatne elemente: prije izbacivanja loših jedinki prvo se izbace eventualni duplikati, dok se od svih jedinki koje imaju isti fitnes (to može nastati primjenom odgovarajuće mutacije) ostavlja samo jedan primjerak. Dalje, ukoliko se kroz definisan broj iteracija provlači najbolja jedinka koja nema zadovoljavajući fitnes (nije dovoljno dobra), tada se i ona odbacuje, a druga po redu jedinka postaje trenutna najbolja. Eksperimenti su pokazali da se ova pojava najčešće dešava u slučaju „nesretnog“ definisanja izrazito loše početne populacije, gdje jedinke budu toliko loše da se mutacijama ne mogu dovoljno popraviti. Sa druge strane, cilj algoritma opravdava i dozvoljava ovakve pojave, te se one i ne smatraju greškom u algoritmu, već samo „lošom inspiracijom“ generatora slučajnih brojeva. Elitistička strategija nije direktno primijenjena (nije unaprijed definisan broj jedinki koje sigurno prolaze u narednu generaciju), ali je pretpostavka da se operatorima mutacije fitnes ne može niti toliko pokvariti niti popraviti, da stare odlične jedinke ne prežive barem do naredne generacije. (Svaka mutacija utiče na promjenu vrijednosti najviše dva intervala.)

Konkretno rješenje

Konkretno rješenje predstavlja ilustraciju prethodno definisanih opštih postavki. Rezultati dobijeni eksperimentom su, po mišljenju autora zadovoljavajući. Ispitivanja su pokazala da se različitim kombinovanjem velikog broja parametara može znatno uticati na samu koncepciju i kvalitet melodije. Na primjer, davanjem veće vrijednosti intervalima mala i velika sekunda (u odnosu na druge) dobijaju se kompozicije čiji su uzastopni tonovi (odnosno intervali) relativno bliski. Najviše su testirane situacije kada se savršeni konsonanti najviše cijene. Po mišljenju autora, u ovom slučaju se dobijaju najljepša rješenja.

U konkretnom rješenju, autor se odlučio za sljedeća ograničenja:

1. Tonovi se uzimaju iz dvije oktave. Postoji mogućnost definisanja skupa „dobrih“ odnosno loših tonova. Po standardnim postavkama, algoritam tonove iz G dura proglašava dobrim, dok se tonovi koji ne pripadaju G dur skali smatraju lošim. Ovo ne znači da se oni potpuno isključuju, već samo da njihova pojava kvare fitnes.
2. Savršenim konsonantima se daje veća vrijednost u odnosu na druge intervale. Vrijednosti intervala su identične vrijednostima iz Tabele 8.
3. Izabrano je da se kompozicija sastoji od četiri takta i ukupno 32 najmanje dužine. Tako svaki takt ima po 8 najmanjih dužina.

4. Unaprijed se definišu referentne aritmetičke sredine i devijacije svakog pojedinačnog takta. U zavisnosti od definisanih sredina i devijacija, dobijaju se različiti rasporedi skladnih i neskladnih intervala. Dosta lijepa rješenja (iako je mišljenje subjektivno) se dobijaju kada se u prvom i četvrtom taktu zahtijeva više savršenih konsonanata, dok se u sredini daje veća sloboda za pojavu drugih intervala.
5. Algoritam je testiran za veličinu populacije od nekoliko desetina (najčešće 30) jedinki. Pokazalo se da je za dobijanje dobrog (često i optimalnog) rješenja dovoljno 100 generacija.
6. Sama rješenja predstavljaju niz tonova različitog trajanja, uz dosta česte pauze. Generalno, algoritam teži pravljenu pauza, jer se tako smanjuje mogućnost pojave loših intervala i tonova, koji mnogo više kvare kvalitet jedinke, nego što ga dobri intervali popravljaju. Stoga dobijene jedinke više zvuče kao dobre improvizacije, nego melodične kompozicije. U krajnjoj liniji, previše su kratke da bi se mogla formirati duža melodija. S obzirom na to, autor je odlučio da dobijene rezultate prezentuje aranžirane u osnovnom aranžmanu, gdje su generisanim jedinkama pridruženi blago podešeni akordi i elementarni ritam bubnjeva kao pratnja.

Implementacija

Algoritam je implemenetiran u programskom jeziku Java i korišten je objektno orijentisani koncept. Za dva osnovna entiteta u algoritmu Jedinka i Populacija realizovane su odgovarajuće klase. Zbog jednostavnosti, svi atributi i metodi su proglašeni javnim. Klasa Populacija sadrži listu Jedinki. Implementirani su neophodni, ali i dodatni atributi i metodi koji olakšavaju rad i omogućavaju bolju funkcionalnost klasa. Spisak ulaznih parametara se podešava preko ulazne datoteke ili direktno u kodu, dok kao rezultat, algoritam daje muzički (tekstualni) zapis koji klase iz paketa JFugue mogu da interpretiraju kao muziku ili midi datoteku koja sadrži kompoziciju.

Samo testiranje dobijenog rezultata (reprodukcija odgovarajuće melodije u zvuk ili notni zapis) vrši se preko glavnog ili pomoćnog programa.

Klasa Jedinka – osnovni elementi

Osnovni atributi

Osnovni atributi klase su:

- Niz u kome se čuva kompozicija (elementi su cjelobrojnog tipa);
- Dužina niza za čuvanje kompozicije;
- Fitnes (tipa double);
- Melodija (tipa String – predstavlja muzički string koju objekti iz paketa JFugue interpretiraju kao muziku).

Klasa sadrži i dodatne attribute koji evidentiraju pozicije tonova, broj različitih tonova, broj tonova po taktovima, intervale i slično i koriste se za bržu realizaciju metoda.

Osnovni metodi

Konstruktori

Klasa ima dva konstruktora. Prvi kreira jedinke na osnovu slučajnog (uz ranije pominjana ograničenja) izbora visina tonova, dok se ritam (raspored trajanja) dobija na osnovu početnih postavki (referentne jedinke). Drugi kreira jedinku na osnovu postojeće, nakon mutacije. Jedinka se ovim konstruktorom kreira u trenutku prenosa parametara (eng. *on the fly*) i ubacuje u listu jedinki populacije.

Klasa sadrži metode za računanje aritmetičke sredine i varijance za svaki takt pojedinačno. Rezultati ovih metoda se (linearno) kombinuju i daju informaciju o odstupanju od unaprijed definisanih aritmetičkih sredina i varijanci referentne jedinke. Ova dva metoda, zajedno sa metodom koji određuje broj „loših“ tonova, u kombinaciji učestvuju u određivanju fitnesa jedinke.

Klasa Populacija – osnovni elementi

Osnovni atributi

Osnovni atribut klase je lista (u p.j. Java ArrayList-a) Jedinki. Izabrana je struktura promjenljive dužine što se jedne strane otežava rad (lakše je raditi sa nizovima nego sa listama), ali se ova struktura praktično koristi prilikom privremenog dodavanja novih (mutiranih) jedinki u populaciju. Stoga se, ovakvim načinom čuvanja jedinki izbjegava potreba za uvođenjem dodatnih „velikih“ objekata.

Pored liste, klasa sadrži i nekoliko pomoćnih atributa koji ubrzavaju rad, kao što su broj dozvoljenih tonova, ukupan broj jedinki itd.

Osnovni metodi

U okviru konstruktora se kreira objekat na osnovu unaprijed zadanih parametara o broju jedinki, broju različitih tonova svake jedinke (za sve jedinke ovaj broj je isti) i rasporedu tonova. Unutar jedne iterativne naredbe se kreiraju pojedinačne jedinke (pomoću prvog od dva navedena konstruktora klase Jedinka). Jedinke se čuvaju u listi redom kako se kreiraju. Nakon kreiranja poziva se metod za sortiranje, gdje se lista jedinki sortira u rastućem redoslijedu po fitnesu. Kako je već rečeno, jedinke koje imaju manji fitnes su bolje, tako da se najbolja jedinka uvijek nalazi na početku liste.

Metod za sortiranje funkcioniše preko pomoćne klase koja implementira standardni javin interfejs za sortiranje Comparator.

Realizovana su tri metoda za tri mutacije, kako je već ranije pomenuto. Svaki od ovih metoda za argument uzima indeks jedinke koja se mutira, a kao rezultat vraća novu jedinku koja nastaje odgovarajućim mutiranjem polazne.

U glavnom metodu za mutiranje, Mutacija(), na svaku jedinku iz prve trećine populacije se pozivaju, sa određenom vjerovatnoćom jedan od tri metoda mutacije. Na svaku jedinku se poziva po unaprijed definisan broj mutacija. Nakon svakog mutiranja u listu jedinki se ubacuje nova jedinka koja je rezultat mutacije. Nakon čitavog procesa mutiranja, lista svih jedinki se sortira.

Metod Selekcija(...) igra značajnu ulogu. Primjenjuje se nakon mutiranja. Prvo se iz kompletne trenutne populacije (koja je zbog mutiranja višestruko veća od propisane) izbacuju jedinke sa istim fitnessom (one po pravilu ne moraju biti jednake). Nakon toga se odlučuje da li je najbolja jedinka prešla starosnu granicu. Ako jeste, tada se i ta jedinka odbacuje. Nakon toga se iz liste izbacuju jedinke sa slabijim fitnessom, sve dok se broj jedinki u listi ne smanji do propisanog.

Glavni program

U glavnom dijelu programa se na osnovu početnih parametara kreira početna populacija. Nakon toga se, u svakoj iteraciji, kreira nova generacija populacije, tako što se primjenjuju operatori mutacije i selekcije. Iterativni postupak se završava kada se ili dostigne tražena tačnost, ili kada se formira maksimalan broj generacija.

Izlazni rezultat je odgovarajući muzički string najbolje jedinke, koji se ili direktno interpretira u zvuk, ili kao tekstualni zapis čuva u izlaznoj datoteci.

Primjeri generisanih jedinki

U ovom poglavlju prezentujemo kompozicije dobijene raznim varijacijama parametara. Analizom parametara i dobijenih kompozicija zaključuje se da se, za zadati skup vrijednosti parametara, dobija odgovarajuća klasa koja sadrži međusobno „slične“ kompozicije.

Primjeri generisanih „main stream⁵⁶“ kompozicija

Za kreiranje kompozicija koje, po subjektivnom mišljenju autora, zadovoljavaju estetske kriterijume određen je odgovarajući skup vrijednosti parametara.

Vrijednosti intervala prikazane su u Tabeli 9. Kako je već rečeno, interval se više cijeni, što je njegova vrijednost manja.

Tabela 9: Konkretno vrijednosti intervala u *main stream* kompozicijama

Intervali	Vrijednosti
prima, čiste kvarte i kvinte, oktava	1
mala i velika terca, mala i velika seksta	2
mala i velika sekunda mala i velika septima	3
svi intervali preko jedne oktave i prekomjerna kvarta	5

Na osnovu podataka iz tabele zaključujemo:

- Savršeni konsonanti su najbolji intervali,
- Terce i sekste se dovoljno cijene da je realna vjerovatnoća da će se i desiti njihova pojava, posebno u drugom i trećem taktu,

⁵⁶ Autor dozvoljava kategorisanje kompozicija po ličnom estetskom doživljaju.

- Sekunde i septime se ne cijene mnogo, te je vjerovatnoća da će se i pojaviti manja nego kod konsonantnih intervala,
- Intervali veći od jedne oktave se izrazito ne cijene.

Za skup „dobrih“ tonova proglašava se skup tonova koji pripadaju G dur ljestvici. Tonovi van G dura se smatraju lošim.

Pošto se kompozicija sastoji od četiri takta, definišu se po četiri referentne vrijednosti za aritmetičku sredinu intervala i varijancu. Vrijednosti su prikazane u Tabeli 10. Algoritam kombinuje podatke iz Tabele 9 i Tabele 10 i tako procjenjuje kvalitet intervala.

Tabela 10: Referentne vrijednosti za aritmetičku sredinu i varijancu

Referentne vrijednosti	Taktovi			
	I takt	II takt	III takt	IV takt
Aritmetička sredina	1	1	1	1
Varijanca	0	0.16	0.24	0

Na osnovu podataka iz Tabele 10 možemo zaključiti:

- Za sva četiri takta zahtijeva se⁵⁷ da intervali budu savršeni konsonanti,
- Eventualne devijacije će se prije desiti u drugom i trećem taktu, nego u prvom i četvrtom.

Treba ponoviti da se ovakvim postavkama ne isključuje pojava i drugih intervala, već se samo smanjuje vjerovatnoća njihove pojave.

Svi težinski faktori koji utiču na fitness su jednaki jedinici.

Na slikama 9-12 su prikazane četiri jedinice dobijene pod navedenim uslovima.



Slika 9: Prva jedinka. Skoro svi intervali su savršeni konsonanti



Slika 10: Druga jedinka. Pojava terci i seksti

⁵⁷ S obzirom da algoritam realizuje konvergenciju proizvoljne kompozicije ka referentnoj, zahtijevanje se može shvatiti kao preporuka



Slika 11: Treća jedinka. Veći broj seksti i terci u drugom, trećem i četvrtom taktu



Slika 12: Četvrta jedinka. Ponovo se javljaju savršeni konsonanti

Specijalni primjerci jedinki

U ovom dijelu pokazujemo kako se promjenom vrijednosti intervala, kao i referentnih vrijednosti za aritmetičku sredinu intervala i varijancu može „upravljati“ komponovanjem.

Jedinka 1. Savršeni konsonanti se najviše cijene (tabela intervala je identična tabeli 9), dok se za referentne vrijednosti postavi zahtjev⁵⁸ da se čitava kompozicija sastoji od intervala koji imaju vrijednost 1 (Tabela 11).

Tabela 11: Referentne vrijednosti za aritmetičku sredinu i varijancu

Referentne vrijednosti	Taktovi			
	I takt	II takt	III takt	IV takt
Aritmetička sredina	1	1	1	1
Varijanca	0	0	0	0

Ostali parametri su isti kao u prethodnom primjeru.

Pod ovakvim uslovima, algoritam je u 47. iteraciji za najbolji rezultat odredio melodiju prikazanu na Slici 13. Fitnes ove kompozicije je nula (optimalno), jer, pored toga što su svi intervali optimalni, kompozicija ne sadrži niti jedan ton van G dura.



Slika 13: Svi intervali su čiste prime, kvarte i kvinte

⁵⁸ Ponovo – zahtjev, tj. preporuka

Jedinka 2. U ovom primjeru sekunde (i male i velike) proglašavamo najboljim intervalima. Vrijednosti intervala prikazane su u tabeli 12. Varijance su jednake onima iz Tabele 11. (Ne dozvoljava se odstupanje od referentnih vrijednosti). Ovo nagovještava da će algoritam težiti da sve intervale postavi na one koji imaju vrijednost 1.

Tabela 12: Vrijednosti intervala gdje se sekunde najviše cijene

Intervali	Vrijednosti
prima, čiste kvarte i kvinte	2
oktava, septime i prekomjerna kvarta	4
mala i velika terca, mala i velika seksta	3
mala i velika sekunde	1
svi intervali preko jedne oktave	5

Algoritam je u 100. iteraciji izbacio melodiju prikazanu na Slici 14. Vidimo dvije interesantne stvari: Algoritam je težio izbacivanju tonova (kompozicija sadrži duge pauze) i zadržan je i „loš“ ton Cis, koji ne pripada ljestvici G dura. Zato je fitnes ove kompozicije veći od nule, te algoritam nije prekinuo iteracije ranije (izvršen je maksimalan broj iteracija i to je bio kriterijum za prekid algoritma). Vidimo da ton Cis koji se javlja jedini utiče na „kvarenje“ fitnesa, te iz toga zaključujemo da bi ova jedinka mogla mutirati u bolju od nje samo ukoliko bi se desila promjena baš tog tona i to još u ton C (bilo koji drugi ton bi narušio interval). Vjerovatnoća da će se ovo desiti je izuzetno mala. Stoga se može pretpostaviti da i nakon dodatnog broja iteracija fitnes jedinki koje nastaju mutacijom ove neće biti bolji. Druga mogućnost jeste da ova jedinka „umre od starosti“, a algoritam pronađe optimalno rješenje zasnovano na nekoj drugoj jedinki.



Slika 14: Svi intervali su male ili velike sekunde

Jedinka 3: Za referentne vrijednosti postavljamo zahtjev⁵⁹ da se čitava kompozicija sastoji od terci, seksti ili oktava. Vrijednosti intervala prikazane su u Tabeli 13, a referentni podaci su ponovo isti kao u Tabeli 11.

⁵⁹ Ibid.



Slika 17: Veća devijacija dozvoljava veću slobodu u intervalima

Jedinka 6: Ako se dozvoli velika devijacija (praktično izbacimo ograničenja), sa istim ostalim parametrima kao kod Jedinke 5, dobija se kompozicija koja zaista nema nikakvog smisla. Ovdje je navodimo samo kao granični slučaj, koji dodatno opravdava kontrolu parametara. Kompozicija je prikazana na Slici 18.



Slika 18: Poremećaj usljed dozvoljavanja velike devijacije

Zaključak

Intenzivan razvoj informaciono komunikacionih tehnologija diktira upotrebu računara i drugih elektornskih uređaja u svim sferama života, pa tako i u umjetnosti. Sa druge strane, kreativnost i sposobnost stvaranja novog i neponovljivog estetski vrijednog sadržaja je isključiva ljudska djelatnost. Određivanje načina i analiza parametara koji računare mogu da usmjere ka uspješnom simuliranju kreativnih ljudskih sposobnosti i konkretno komponovanju muzike, su danas predmet izučavanja naučne oblasti koja se nalazi na tromeđi matematike, informatike i teorije muzike.

Muzika je od antičkih civilizacija bila predmet izučavanja matematičara. Još u Staroj Grčkoj su, u vremenu Pitagore, prepoznate matematičke pravilnosti koje opisuju „skladne intervale“. Pokušaji formalizovanja muzike i procesa komponovanja nisu zaobišli ni slavne kompozitore kao što su Bah, Hajdn ili Mocart. Značajan korak ka formalizovanju muzike i „kroćenju“ beskonačnog prostora svih mogućih tonova predstavlja uvođenje dvanaest tonske jednako temperirane skale. U novije vrijeme, od sredine dvadesetog vijeka, upotrebom računara povećava se i broj pokušaja upotrebe algoritama u komponovanju muzike. Razvoj elektronskih muzičkih instrumenata diktira i razvoj i sintezu novih zvukova, što je opet zasebna oblast nauke, u kojoj pored matematike, teorije muzike i računarstva, značajnu ulogu igraju naučne oblasti fizike i elektrotehnike.

S obzirom na svoju prirodu, genetički algoritmi predstavljaju dobar način za pretraživanje ogromnog i neuređenog muzičkog prostora mogućih kompozicija. Uzastopna primjena genetičkih operatora mutacije i selekcije iz iteracije u iteraciju omogućavaju kontinuirano „poboljšanje“ dobijenih rezultata. Operator ukrštanja, u kombinaciji sa selekcijom simulira inovativni proces stalnog dobijanja novih rješenja, koja predstavljaju kombinaciju prethodnih, uzimajući njihove najbolje dijelove.

U ovom radu je napravljen pregled upotrebe genetičkih algoritama u muzici, sa naglaskom na komponovanje muzike upotrebom ovih algoritama. Kao ilustraciju upotrebe genetičkih algoritama u komponovanju, autor je razvio jedinstven, sopstveni genetički algoritam koji kontrolom nekih „mjerljivih“ elemenata muzičke kompozicije daje, po subjektivnom estetskom osjećaju autora, zadovoljavajuće rezultate. Rezultati (kompozicije) koji se mogu dobiti iznesenim algoritmom po mišljenju autora, zadovoljavaju i neke objektivne kriterijume „lijepo“ kompozicije: dobijene kompozicije sadrže intervale koji su prijatni ljudskom uhu, kompozicije imaju smislen ritam, a uz blago prilagođen aranžman i odgovarajuću pratnju zvuče neobično, ali prijatno.

Sa aspekta praktičnog korištenja, ovaj algoritam daje mogućnost upravljanja raznim parametrima koji utiču na kvalitet i oblik kompozicije. Postojanje referentne jedinice (ili unaprijed zadatih parametara) ubrzava proces odabira i dobijanja ritmične i relativno skladne kompozicije. Striktno vezivanje rezultata algoritma za referentne vrijednosti za posljedicu rezultuje pojavu velikog broja „sličnih“ kompozicija. Podešavanjem vrijednosti intervala, ali i parametara koji mijenjaju uticaj sličnosti sa referentnom jedinkom, omogućava se pretraživanje (eksploatacija) skupa svih mogućih kompozicija. Ovo, prema mišljenju mnogih autora (kako je već navedeno u [2]), predstavlja dobar način za dobijanje

zadovoljavajućeg rješenja. Dio prostora svih dozvoljenih kompozicija se tako može klasifikovati na različite tipove kompozicija, čiji se predstavnici dobijaju ovim algoritmom.

Kodiranje kompozicije preko niza tonova i pauza (uz dodatne informacije o dužini trajanja) omogućava efikasno i brzo upravljanje samom kompozicijom, tonovima i njenim ritmom, kao i primjenu odgovarajućih matematičkih funkcija koje tonovima, intervalima i drugim „muzičkim“ parametrima daju brojne vrijednosti na kojima se mogu vršiti aritmetičke i logičke operacije, neophodne za funkcionisanje bilo kakvih algoritma.

Ako se algoritmu postave „teški“ kriterijumi, gdje se fitnes kompozicije teško popravlja mutiranjem pojedinačnih tonova, algoritam će težiti praviljenju pauza i dobijene kompozicije će više ličiti na improvizacije, nego na kompozicije sa prepoznatljivom dužom melodijom. Sa druge strane, ako se dozvoli veća varijanca (odstupanje od propisanih intervala), tada algoritam prije dostizanja maksimalnog broja koraka predlaže rješenje koje je (prema zadatim kriterijumima) optimalno. Balansiranje svim parametrima koji utiču na kvalitet i tip kompozicije predstavlja vještinu koja se može steći detaljnom analizom svih elemenata algoritma. U radu su na konkretnim primjerima prikazani način upravljanja parametrima, kao i dobijene kompozicije.

Direktno uključivanje ljudskog faktora u odabir melodija ovim algoritmom nije predviđeno. Sa druge strane, brzo izvršenje ovog algoritma doprinosi kreiranju velikog broja potencijalno dobrih rješenja, od kojih su samo neka predstavljena u ovom radu. Praktično, svakim pokretanjem algoritma dobija se potencijalno rješenje, koje je algoritam proglasio kao najbolje moguće, u zavisnosti od „raspoloženja“ generatora slučajnih brojeva. Povećanjem broja iteracija se može dobiti bolje rješenje, iako to u opštem slučaju, kako je ranije analizirano, nije obavezno.

Rad sa dužim kompozicijama (u prezentovanom algoritmu rađeno je sa kompozicijama dužine četiri takta) je u ovakvom algoritmu teorijski moguć. Implementacija dodatnih faktora koji utiču na fitnes (analiza trajanja tonova, broj sinkopa, gustina tonova i slično) bi uz uključivanje operatora ukrštanja omogućili dobijanje još kvalitetnijih kompozicija. Stoga, kao prvu nadogradnju ovog algoritma autor predlaže upravo implementaciju dodatnih kriterijuma za računanje fitnesa, odgovarajuće mutacije koje mijenjaju fitnes u skladu sa tim kriterijumima, kao i implementaciju operatora ukrštanja, ukoliko je cilj upravljanje dužim kompozicijama.

Literatura

- [1] Andrew Gartland-Jones, Peter Copley, *The Suitability of Genetic Algorithms for Musical Composition*, Contemporary Music Review, VOL. 22, No. 3, 43–55, 2003.
- [2] Biles, J. A. *GenJam: A genetic algorithm for generating jazz solos*. In ICMC Proceedings 1994. The Computer Music Association, 1994.
- [3] Goldberg, D. E., *The Design of Innovation*, Dordrecht: Kluwer Academic Publishers, 2002.
- [4] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, *Introduction to algorithms*, MIT Press, 2001
- [5] Victor J. Katz, *A History of Mathematics, An Introduction*, Addison – Wesley, 1998.
- [6] David Cope, *The algorithmic composer*, Madison, WI: A-R Editions Inc., 2000.
- [7] M. P. Ferreira: *Proportions in Ancient and Medieval Music*, Grad Assayag, Hans G. Feichtinger, Jos-Francisco Rodrigues (editors): Mathematics and music, European Mathematical Society, Springer 2002
- [8] Grout, Donald Jay and Claude V. Palisca, *A History of Western Music*. 5th ed. W. W. Norton & Company: New York. 843 pp, 1996.
- [9] Madarász Sz. Rozália: *Matematika i muzika*, Departman za matematiku i informatiku, PMF Novi Sad, 2009.
- [10] The Grove Concise Dictionary of Music, edited by Stanley Sadie © Macmillan Press Ltd., London.
- [11] John F. Putz: *The Golden Section and the Piano Sonatas of Mozart*, Mathematics Magazine, Vol.68, No. 4. pp. 275-282, Oct., 1995.
- [12] John Chuang, *Mozart's Musikalisches Würfelspiel*, Internet članak, 1995.
- [13] John A. Maurer, *A Brief History of Algorithmic Composition*, Internet članak, 1999.
- [14] John A. Biles, *Evolutionary Computation for Musical Tasks*, Evolutionary Computer Music (Eduardo Reck Miranda and John Al Biles (Eds)), Springer, 2007
- [15] George Papadopoulos; Geraint Wiggins, *AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects*, In AISB Symposium on Musical Creativity, 1999.
- [16] Kratica, J. *Paralelizacija genetskih algoritama za rešavanje nekih NP – kompletnih problema*, doktorska disertacija, Matematički fakultet Beograd, 2000.
- [17] Ender Özcan, Türker Erçal, *A Genetic Algorithm for Generating Improvised Music*, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, Volume 4926, 2008.

- [18] Geraint Wiggins, George Papadopoulos, Somnuk Phon-Amnuaisuk, Andrew Tuson, *Evolutionary Methods for musical composition*, In Proceedings of the CASYS98 Workshop on Anticipation, Music and Cognition Liege, Belgium, 1998.
- [19] Michael Towsey, Andrew Brown, Susan Wright and Joachim Diederich, *Towards Melodic Extension Using Genetic Algorithms*, Educational Technology & Society 4 (2) 2001
- [20] Goldberg, D.E. *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. Kluwer Academic, Boston. 2002.
- [21] Rodney Waschka, *Composing with Genetic Algorithms: GenDash*, Evolutionary Computer Music (Eduardo Reck Miranda and John Al Biles (Eds)), Springer, 2007.
- [22] John Al Biles, *Improvising with Genetic Algorithms: GenJam*, Evolutionary Computer Music (Eduardo Reck Miranda and John Al Biles (Eds)), Springer, 2007.
- [23] Andrew Gartland-Jones, *Can a Genetic Algorithm Think Like a Composer?*, Generative Art, 2002.
- [24] Eduardo Reck Miranda, *Computer Sound Design, Synthesis techniques and programming*, (second edition) Focal Press, 2002.
- [25] Phil Husbands, Peter Copley, Alice Eldridge, James Mandelis, *An Introduction to Evolutionary Computing for Musicians*, Evolutionary Computer Music (Eduardo Reck Miranda and John Al Biles (Eds)), Springer, 2007.
- [26] Ricardo A. Garcia, *Growing Sound Synthesizers using Evolutionary Methods*, In E. Bilotta, E. R. Miranda, P. Pantano, and P. M. Todd (Eds.), Proceedings of ALMMA 2001 Workshop on Artificial Life Models for Musical Applications, Cosenza, Italy. Editoriale Bios, pp. 99–107, 2001.
- [27] Koza, J. R., Bennett III, F. H., Andre, D., Keane, M. A., Dunlap, F. *Automated Synthesis of Analog Electrical Circuits by Means of Genetic Programming*. IEEE Transactions on Evolutionary Computation, Vol. 1, No. 2. 1997.
- [28] Dave Koelle, *The Complete Guide to JFugue*, Programming Musin in Java, 2008.