



IEM Report 41/07:

**Zur Modellierung von Kernmelodien im Rahmen des
„Virtual Gamelan Graz (VGG)“**

Verfasser:

Gerhard Nierhaus

Kooperationspartner:

Institut für Musikethnologie (IME)

Zukunftsfonds Steiermark (GZ: A3-25R9-05/19)

November, 2007

Inhaltsverzeichnis

Aufgabenstellung	3
Zum Begriff der Kernmelodie	3
Notationsformen	3
Kodierung	4
Analysen	5
Generierungsalgorithmen.....	5
1. Anforderungen	5
2. Etablierte Modelle.....	6
Entwicklung eines Generierungsmodells.....	8
Schema.....	9
Experimente und Evaluierungen anhand der Generierung von Cantus Firmi	10
Generierung von Kernmelodien.....	11
Wahl des Korpus.....	11
Generierung und Klassifizierung von Gongperioden und Einleitungssphrasen....	12
Generierung von Kernmelodien.....	12
Ausblick	13
Literatur.....	14

Aufgabenstellung

Als eine Aufgabenstellung im Rahmen des „Virtual Gamelan Graz (VGG)“ sollte ein Ansatz für die automatische Generierung von Kernmelodien entwickelt werden. Prinzipiell können bei der Modellierung eines musikalischen Stils zwei Zugänge gewählt werden. Wissensbasierte Methoden verwenden Regeln und Rahmenbedingungen um stilkonformes musikalisches Material zu generieren. Die zweite Herangehensweise, die für die Entwicklung eigener Strategien gewählt wurde, benötigt kein fachspezifisches Wissen über die zu modellierende Domäne. Eine wichtige Voraussetzung ist jedoch ein ausreichend großer Bestand stilkonformer Kompositionen. Als Korpus stand eine umfangreiche Datenbank mit Kernmelodien unterschiedlicher Stile zur Verfügung. Für die Weiterverarbeitung der Daten wurden verschiedene Kodierungsstrategien entwickelt. Eine mehrdimensionale Repräsentationsform erlaubte umfangreiche Analysen, die als Basis für die weitere Entwicklung eines Generierungsalgorithmus erstellt wurden. Die Effizienz des Algorithmus wurde durch die Modellierung von Musikstilen erprobt, denen ein bekanntes Regelwerk übergeordnet ist. Die Generierung von Kernmelodien erfolgte schließlich in einem zweistufigen Prozess: Es wurden großformatige Formbestandteile generiert, klassifiziert und aufgrund ihrer Eigenschaften zu neuen Kernmelodien kombiniert. Erste Überprüfungen der Ergebnisse ergaben gute Resultate, und eröffnen interessante Möglichkeiten weiterer wissenschaftlicher Forschung.

Zum Begriff der Kernmelodie

„Balungan“ (jav.) ist die Bezeichnung für ein melodisches Gerüst, bzw. die sogenannte „Kernmelodie“ der zentraljavanischen Gamelanmusik: „The 'melody' recorded in collections of notation for the gamelan is called the *balungan*, meaning 'skeleton' or 'framework' – it is also the word for the structural framework of a building – or sometimes *balunganging gendhing*. The term correctly suggests that what is notated is only part of the picture.“[35, S. 23]. Die letzte Bemerkung von Richard Pickvance weist darauf hin, dass die Kernmelodie mehr als ein abstraktes Gerüst begriffen wird, welches erst durch eine mehr oder minder elaborierte Ausgestaltung durch die Instrumente des Gamelanorchesters musikalisch in Erscheinung tritt.

Notationsformen

Neben der im 19. Jh. entstandenen Notationsformen *ranté* („Kettennotation“) und *ândhâ* („Leiternnotation“) wird vor allem die in den 1920-er Jahren entstandene Notation des *kepatihan* („Ziffernotation“) häufig für eine Repräsentation der Kernmelodie verwendet. Während *ranté* und *ândhâ* die Tonstufen der Kernmelodien auf Linien in horizontaler und vertikaler Leseweise darstellen, erfolgt in *kepatihan* eine Repräsentation durch Ziffern.

Im *laras sléndro*, das durch eine fünfstufige Unterteilung der Oktave gekennzeichnet ist, werden hier die Ziffern 1, 2, 3, 5 und 6 zur Repräsentation der Tonhöhen verwendet. Im siebenstufigen *laras pélog* hingegen werden die einzelnen Tonstufen durch die Ziffern von 1 bis 7 angegeben. Zusätzliche Symbole können Oktavlagen, als auch bestimmte Instrumente des Ensembles bezeichnen. In der dem Projekt zur

Verfügung stehenden Datenbank von Barry Drummond [18], die eine umfassende Sammlung von Kernmelodien darstellt, werden in einem internen Format die Tonstufen ausserdem durch Buchstaben dargestellt, die als Information auch die aktuelle Oktavlage miteinbeziehen. Weitere Symbole bezeichnen metrische Schwerpunkte durch die Angabe bestimmter Instrumente. *Gatras*, endgewichtete 4-er Gruppen, die in einer vorsichtigen Analogie zum Begriff des „Taktes“ in der abendländischen Musik gesehen werden können, sind durch Leerstellen zwischen den Zeichenfolgen ersichtlich gemacht.

Kodierung

Um die Informationen der „Buchstabennotation“ einer informatischen Verarbeitung zugänglich zu machen wurde ein Notationsparser entwickelt, der die Informationen der Kernmelodien, wie Tonhöhe und Dauer als einen Vektor von Vektoren darstellt. Dieser einfache Parser wurde am IME durch Rainer Schuetz und Florian Rohrer weiterentwickelt um ein Datenformat für die weitere Verarbeitung von Kernmelodien bereit zu stellen.

Auf eine Anregung Robert Höldrichs wurde eine weitere mehrdimensionale Repräsentationsform entwickelt, welche die Möglichkeiten einer informatisch-musikalischen Analyse von Kernmelodien beträchtlich erweitert. In diesem Modell werden die Informationen als eine Liste von 7-dimensionalen Vektoren dargestellt, die jeweils einen gleich langen Abschnitt der Kernmelodie repräsentieren. Die Länge dieses Abschnitts kann vom Benutzer angegeben werden und wird sinnvollerweise in Bezug auf metrische Struktur der Kernmelodien gewählt – so wird z.B. bei einer gewünschten Darstellung von vier Schlägen pro gatra ein Wert von 1 angegeben. (aus 0.5 würden demgemäß acht Einheiten pro gatra kodiert etc.)

Folgende Abbildung zeigt oben die „Buchstabennotation“ einer Gongperiode¹, unten eine Darstellung des ersten gatras in Form mehrdimensionaler Vektoren - kleinste Einheit: 1.

IJLM -LJI) -II-^ GFGI) -IKJ^ INMN) LMNL^ MNML@
 [[9, 2, 2, 1, 0, 0, 1], [10, 3, 2, 1, 0, 0, 2], [12, 5, 2, 1, 0, 0, 3], [13, 6, 2, 2, 0, 1, 4] ...

Abb. 1: „Buchstabennotation“ (oben), mehrdimensionales Repräsentationsverfahren (unten).

Die Dimensionen der einzelnen Vektoren stellen folgende Informationen bereit:
 (0) Tonhöhe absolut, (1) Tonhöhe relativ, (2) Oktavlage, (3) Länge des Tons in dem man sich befindet, (4) Überbindung von - bzw. Wie viel gleiche Tonhöhen sind dem aktuellen Ton vorausgegangen) , (5) Überbindung bis - bzw. Wie viel gleiche Tonhöhen folgen, (6) Position im Gatra in Bezug auf kleinste rhythmische Einheit.
 Da die Dimensionen der Vektoren einzeln oder in jeder beliebigen Kombination abgefragt werden können, ermöglichen sich unterschiedliche Sichtweisen auf das musikalische Material.

¹ Ein „gongan“ oder eine Gongperiode bildet einen Grundbaustein einer Kernmelodie. Eine Kernmelodie besteht aus einer oder mehreren Gongperioden mit einer obligaten „buka“, einer Einleitungsformel.

Analysen

Mit Hilfe des zuvor genannten mehrdimensionalen Repräsentationsverfahren konnten Kernmelodien einer Reihe von Analysen im Hinblick auf einzelne oder kombinierte musikalische Parameter unterzogen werden. Da die Gongperioden der untersuchten Kernmelodien immer eine bestimmte Anzahl von gatrās aufweisen, wurden weitere Analysefunktionen entwickelt, um mögliche „dominante“ Positionen innerhalb von Kernmelodien zu erkennen. Diese Funktionen untersuchen Häufigkeiten verschiedener musikalischer Parameter und kennzeichnen auffällige Positionen im Bereich eines vom Benutzer vorgegebenen Abschnitts². Fragestellungen, die mit diesen Analysefunktionen untersucht werden können sind unter anderem: Gibt es bestimmte Positionen in einem Gatrā, einer Gongperiode etc. an denen ein eingeschränkter Tonhöhenvorrat verwendet wird? Gibt es Positionen in einem Abschnitt von vier gatrās an denen eine große Bandbreite verschiedener Oktavlagen möglich ist? Im Kontext dieser und ähnlicher Fragestellungen wurden auch erweiterte Analysefunktionen entwickelt, die auch die Erkennung auffälliger Positionskombinationen ermöglichen.

Einige der zuvor genannten nichtwissensbasierten Analysemethoden ermittelten Auffälligkeiten in der Struktur von Kernmelodien, die auch von Rainer Schuetz als der Musikethnologie bekannte Beobachtungen bestätigt werden konnten.

Hauptziel der Analysen war jedoch die Beschaffenheit des Korpus im Hinblick auf bestimmte Anwendung oder Entwicklung von Algorithmen zur Genese von neuen und dennoch „stiltypischen“ Kernmelodien zu untersuchen, bzw. in weiterer Folge ein zusätzliches Instrument zur Evaluierung von Neugenerierungen bereit zu stellen.

Generierungsalgorithmen

1. Anforderungen

Die wesentlichen Anforderungen, die sich aus den oben genannten Beobachtungen und den Projektvorgaben für einen Generierungsalgorithmus ergaben waren:

- Die Generierungsmächtigkeit des Algorithmus soll sich, als eine Anwendung eines nichtwissensbasierten Systems, primär aus der Beschaffenheit und der Struktur des Korpus ergeben.
- Der Algorithmus soll optimal für die symbolische Verarbeitung eindimensionaler Zeichenketten geeignet sein, die in ihrem Aufbau über eine große Kontextsensitivität verfügen.

² Für diesen Abschnitt – z.B. vier metrische Einheiten – wird auch ein Parameter angegeben, der die „Weiterrückung“ angibt – z.B. ebenfalls vier metrische Einheiten. In diesem Fall würden alle aufeinanderfolgenden Gruppen von vier metrischen Einheiten in Bezug auf Häufigkeiten anzugebender musikalischer Parameter miteinander verglichen. Ein mögliches Resultat könnte z.B. ergeben, dass an der ersten Stelle aller – der jeweils vier metrischen Positionen betragenden – untersuchten Einheiten der musikalische Parameter X oder die Kombination der Parameter X, Y, Z nur innerhalb einer kleineren Bandbreite auftritt. Wenn sich – optional – die Parameter für Abschnittsgröße und Weiterrückung unterscheiden, können dadurch z.B. auch Untersuchungen an Übergängen betrachteter Abschnitte durchgeführt werden.

- Als Anforderung in Bezug auf einen reibungslosen workflow im Rahmen des Projekts sollte die Implementierung ausserdem in der Computermusiksprache SuperCollider erfolgen.

2. Etablierte Modelle

Im Bereich der nichtwissensbasierten Systeme im Rahmen der algorithmischen Komposition werden einige Algorithmenklassen prominent für die Erstellung stilkonformen musikalischen Materials verwendet. Markov- und Hidden-Markov Modelle [1, 9, 17, 20], generative Grammatiken [2, 3, 4, 5, 8, 11, 22, 31, 38, 39, 40] Transitionsnetze [12, 13, 14, 15, 16], genetische Algorithmen [7, 10, 26, 30, 32, 33] künstliche neuronale Netze [6, 21, 27, 41] und verschiedene Anwendungen aus dem Bereich der künstlichen Intelligenz [19, 34, 36, 37, 42, 43], wie case-base reasoning oder maschinelles Lernen ermöglichen unter anderem die Genese von stiltypischen musikalischen Ausgaben auf der Basis eines zugrundeliegenden Korpus. Für die konkreten Anforderungen des Projekts erwiesen sich viele der genannten Algorithmenklassen als unzureichend:

Markov-Modelle sind durch deren einmal gewählte Ordnung³ auf die Behandlung einer bestimmten Kontexttiefe beschränkt. Wenn – um eine hohe Kontextsensitivität zu erhalten – Modelle hoher Ordnung für die Genese verwendet werden, besteht die Gefahr, dass hauptsächlich große Abschnitte des Korpus neu generiert werden, ausserdem werden bei diesem Zugang die in Modellen niedriger Ordnung enthaltenen Informationen verschleiert.

Genetische Algorithmen sowie auch neuronale Netze können zwar interessante Ergebnisse hervorbringen, erweisen sich jedoch im Allgemeinen bei der Behandlung eines weiten Kontexts als nachteilig.

Zudem verlangen genetische Algorithmen, neuronale Netze, case-based reasoning Systeme so wie auch viele Anwendungen des maschinellen Lernens Evaluierungen durch einen Benutzer oder einer algorithmische Fitnessfunktion um die Qualität der Ergebnisse im Laufe eines Trainingszyklus zu verbessern. Eine algorithmische Bewertung von Generierungen ist zumeist durch ein kodiertes komplexes Regelwerk zu erreichen, das im Falle der Kernmelodien nicht zur Verfügung steht. Ein Training eines lernfähigen Systems zur Produktion von Kernmelodien wäre nur mit einem enormen persönlichen Aufwand eines Experten realisierbar und wurde für dieses Projekt auch aufgrund anderer Überlegungen nicht in Betracht gezogen.

Generative Grammatiken, die zumeist in Anlehnung an die von Noam Chomsky entwickelte Hierarchie im Rahmen der algorithmischen Komposition verwendet werden, sind kompakt formulierbare Ersetzungssysteme - wie auch der verwandte Formalismus der Transitionsnetze – und erlauben überdies die Verarbeitung eines beliebig weiten Kontexts. Generative Grammatiken können sowohl wissensbasiert – als komplex formulierbare Regelsysteme – als auch nichtwissensbasiert – in Form der grammatikalischen Inferenz [24, 25, 28, 29] zur Generierung musikalischen Materials eingesetzt werden. Generative Grammatiken wurden auch als Instrument der musikethnologischen Forschung eingesetzt [3, 4, 5, 8, 31]. Im Kontext dieses Projekts sind auch aus historischen Gründen zwei prominente Ansätze erwähnenswert, die sich mit der Formulierung eines Regelwerks im Rahmen verschiedener Stile des Gamelan

³ Die Ordnung eines Markov-Modells gibt eine bestimmte Anzahl von sequentiell aufeinander folgenden Symbolen an, die für eine Berechnung von Übergangswahrscheinlichkeiten für die Generierung eines darauf folgenden Symbols verwendet werden - diese einmal determinierte Kontexttiefe entspricht meist aber nicht den strukturellen Gegebenheiten musikalischen Materials.

auseinandersetzen. Judith und Alton Becker erstellen „A grammar for the musical genre Srepegan“ [2] – ein einfaches grammatisches Modell basierend auf einem Korpus von sieben Kernmelodien. David Hughes [22] erweitert den Geltungsbereich seines Ansatzes auf drei Stile und bedient sich im Gegensatz zu Beckers formaler Kodierung natürlichsprachlicher Mittel zur Beschreibung seines Regelwerks.

Einen wesentlichen Beitrag zur Erschließung eines genuinen Musikstils leisten Jim Kippen und Bernard Bel⁴ in den 1980-er Jahren mit ihrem grammatikalischen Inferenzmodul „QUAVID“, das in einem diskursiven Prozess der Generierung und Evaluierung Grammatiken für die Produktion von „quaida’s“ einem Genre nordindischer Tablamusik generiert.

Einen Ansatz, der von den bisher genannten Arbeiten am ehesten den konkreten Anforderungen an die Modellierung entsprach entwickelt Teuvo Kohonens mit seinem „Dynamically Expanding Context“ [24]. Als eine „Self-Learning Musical Grammar“ [25] erlaubt der Algorithmus die Erfassung eines variablen Kontexts innerhalb eines musikalischen Korpus und erstellt eine kontextfreie Grammatik zur Produktion neuen Materials.

Wenn die Ersetzungsregeln einer generativen Grammatik einen Korpus genau beschreiben, bzw. denselben neu generieren sollen muss jeder Nachfolger durch seinen vorhergehenden Kontext genau definiert sein. Wenn zum Beispiel⁵ aufgrund einer Symbolkette von: ABCDEFG...IKFH...LEFJ... ein konkreter Nachfolger von F ermittelt werden soll, ergeben sich in Abhängigkeit der betrachteten Kontexttiefe unterschiedliche Möglichkeiten der Ersetzung. Kontexttiefe 1 (F) ergibt als Nachfolger G, H und J - es besteht also keine Möglichkeit allein aufgrund des Vorgängers F einen exakten Nachfolger zu bestimmen. Bei Erweiterung des Kontexttiefe um ein Symbol (EF, KF, EF) kann für KF der exakte Nachfolger H bestimmt werden, für EF ergeben sich die alternativen Nachfolger G und J. Erst bei einer Kontexttiefe von 3 (DEF, LEF) können die konkreten Nachfolger G (Kontext: DEF) und J (Kontext: LEF) ermittelt werden. Das Beispiel zeigt, dass zur Produktion eines bestimmten nachfolgenden Symbols in einem Korpus unterschiedliche Kontexttiefen benötigt werden - eine Gegebenheit, die z.B. durch ein Markow-Modell aufgrund der einmal festgelegten Ordnung, resp. Kontexttiefe nicht modelliert werden kann. Kohonens Ansatz stellt gleichsam ein Markow-Modell variabler Ordnung dar, das für jedes Symbol der Eingabe die notwendige Kontexttiefen - die Anzahl der vorangegangenen Zeichen - in der Gestalt von Ersetzungsregeln formuliert, welche die Abfolge der Symbole im Korpus erschöpfend beschreiben können. Der Algorithmus liest sukzessive die Zeichen der Eingabe und erstellt Regeln der Art Vorgänger -> Nachfolger. Wenn ein Vorgängersymbol erneut als Eingabe vorliegt und einen anderen Nachfolger bestimmt, wird die zuvor erstellte Regel mit einem sogenannten Konfliktbit belegt und der Kontext solange erweitert, bis kein Konflikt in der Anwendung der Regeln mehr besteht, z.B. KF -> H, DEF -> G, LEF -> J. Die erstellten Regeln lassen sich auch durch eine Baumstruktur⁶, wie in Abbildung X ersichtlich wiedergeben.

⁴ Die von Kippen und Bel entwickelte Software „Bol Processor“ ist eine umfassende Umgebung für algorithmische Komposition im Rahmen generativer Grammatiken. Das System wird ständig weiterentwickelt und wurde vor der Entscheidung für die Computermusiksprache SuperCollider für die Modellierung grammatikalischer Modelle in Erwägung gezogen.

⁵ Aus: [24].

⁶ Eine solche Baumstruktur ist auch als *Lempel-Ziv Baum* im Kontext eines Datenkompressionsverfahrens bekannt, das immer nur die jeweils kürzesten Zeichenketten, die noch nicht erkannt wurden neu kodiert, vgl. [44].

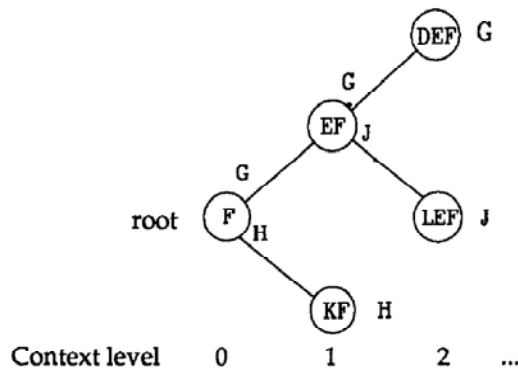


Abb 2: Darstellung variabler Kontexttiefen innerhalb einer Baumstruktur; [25].

Für die Produktion neuen Materials wird nun ein „depth parameter“ angegeben, der bestimmt um wie viele Schritte der „optimale“ Kontext eines jeden Symbols vermindert wird. Es werden also Regeln zur Anwendung gebracht, die zuvor mit einem Konfliktbit belegt waren - im konkreten Fall können zB. Bei einem depth parameter von 1 also auch Symbolfolgen der Art DEF -> J, bzw. LEF -> G erzeugt werden, die in der Symbolfolge der Eingabe nicht vorkommen.

Entwicklung eines Generierungsmodells

Im Hinblick auf die Anforderungen des Projekts wurde ein Algorithmus entwickelt, der ebenfalls die Verarbeitung eines variablen Kontexts erlaubt, jedoch in seiner „Ordnung“ prinzipiell unbeschränkt ist und ausserdem die Übergangswahrscheinlichkeiten der Symbolfolgen im Korpus reflektiert. Als eine zusätzliche Erweiterung ermöglicht ein „Backtracking Parameter“ die Angabe einer minimal notwendigen Kontexttiefe für die Generierung eines jeden neuen Zeichens.

Im Unterschied zu Kohonens „Self Learning Grammar“ werden keine Ersetzungsregeln erzeugt, sondern der Algorithmus durchschreitet gleichsam bei jeder Neugenerierung Bereiche eines gegebenen Korpus, wobei der zurückgelegte Weg gleichzeitig die Ausgabe bildet. Der Algorithmus wird mit einem Symbol oder einer Sequenz von Symbolen als Predecessor initialisiert und ermittelt aus dem Korpus die entsprechenden Symbole als Successors. Predecessor und Successor werden im nächsten Schritt zu einem neuen Predecessor und der Prozess beginnt von neuem. Auf diese Weise wird die Kontexttiefe solange erweitert bis sich nur mehr ein möglicher Successor ergibt. Durch Backtracking wird nun die Kontexttiefe so lange verringert, bis wieder alternative Möglichkeiten der Fortsetzung bestehen. Die musikalischen Beispiele des Korpus können hintereinander in eine Liste kopiert werden, wobei sich eine Kennzeichnung von Terminalen, die jeweils am Anfang und am Ende eines jeden Beispiels stehen empfiehlt. Die Kennzeichnung des Endes eines jeden Beispiels ermöglicht dem Algorithmus den Abbruch der Generierung und die Ausgabe des Resultats. Die Kennzeichnung des Anfangs ist obligat ermöglicht aber bei einer entsprechenden Eingabe mit einem Predecessor zu beginnen, der auch am Anfang eines Beispiels des Korpus steht. Da sich der Algorithmus wie eine Schlange verhält, die immer mehr Material in sich hineinschlingt, aber ganz in der Manier eines Feinschmeckers auf einer Auswahl von mindestens zwei alternativen „Häppchen“ besteht, wurde dieses Verfahren als „Context Snake“ bezeichnet. Der „Backtracking Parameter“ gibt die minimal notwendige Anzahl von Symbolen des Predecessors an. Sollte diese Anzahl unterschritten werden, werden solange rückwirkend alternative

Verzweigungen gewählt, bis schließlich – ausgenommen natürlich am Beginn - jeder Übergang der Neugenerierung die minimal notwendige Kontexttiefe aufweist. Im Vergleich zu Kohonens Ansatz ermöglicht der „Backtracking Parameter“ eine sehr gute Anpassungsfähigkeit an die Gegebenheiten des Korpus und die Kriterien für die Ausgabe. Als ein weiterer Vorteil ist die Beachtung der Übergangswahrscheinlichkeiten im Korpus zu sehen. Im Vergleich zu einem herkömmlichen Parsing einer Grammatik ergibt sich jedoch durch das umfangreiche Backtracking bei jeder Neugenerierung eine wesentlich erhöhte Berechnungsdauer.

Schema

Folgendes Schema illustriert die Funktionsweise des Algorithmus; die Symbole bezeichnen in diesem Beispiel musikalische Terminale, die je nach Kodierung für die Repräsentation einzelner oder mehrerer musikalischer Parameter, verwendet werden (in diesem Fall multidimensionale Vektoren, siehe oben).

1. Gib einen Predecessor (P) an, von dem aus die Suche gestartet werden soll;
P wird erstes Element der Liste „Result“ (R)
2. Erzeuge eine leere Liste „Auswahl“ (A)
3. Ermittle alle Successors (S) von P aus dem Korpus.
Jedes P+S wird zu einem neuen P
Jedes P wird Element von A
If Anzahl der Elemente in A = 1, weiter bei Punkt 4
If Anzahl der Elemente in A > 1 weiter bei Punkt 3
4. Wähle zufällig ein P aus A und übergebe letztes Symbol dieses P an R
if letztes Symbol von P = Symbol für Ende eines Abschnitts, weiter bei Punkt 5
Ansonsten: weiter bei Punkt 2
5. if letztes Symbol von P = Symbol für Ende eines Abschnitts, übergebe dieses Symbol an R, weiter bei Punkt 5
Ansonsten: Entferne erstes Symbol von P und gehe weiter bei Punkt 2
6. Gib R aus / ENDE

Die Funktionsweise des Algorithmus lässt sich anschaulich durch Textgenerierungen demonstrieren. Als Korpus für die folgenden Generierungen dienten die Strophen des „Erlkoenigs“ von Goethe – bei jeder neuen Strophe wird der Kontext Parameter um den Wert 4 erhöht:

Meine Tochter schein Nanch blkoenig nicht!
Meine Tochter fuhig, br,
Meine Mut so spaeund Wer, mein Vater, und hat mir!
Gestanzend Spieleibeschweine so grau.

Meine Mutter grauselt deine siebes gettern und talt!
Dene Gest dein Knabe ruhig, bleist den Hof mich an den nicht!

Den Knabe die Gester fasst ihn schoene Sohn, er haelt in dem Stren dich eschoene
Gester, mein Vater, und haelt ihn war tot.

Ich liebes Kind, komm geh' mit mir!
Gar schoene Spiel ich seh'es getan.

Wer reitet so spaet durch Nacht und Wind?
Es ist ein Nebelstreif.

Ich lieb dich, mich reizt deine schoene Gestalt,
Und bist du nicht,
Was Erlenkoenig mir leise verspricht?
Sei ruhig, mein Kind,
In duerren Blaettern saeuselt der Wind.

Experimente und Evaluierungen anhand der Generierung von Cantus Firmi

Experimente mit diesem Algorithmus⁷ zeigten unter anderem gute Ergebnisse in Bezug auf die Generierung von Chormelodien. Für den Korpus werden einige dorische Melodien gemäß den Regeln Knud Jeppesens verwendet. In seinem Lehrbuch der klassischen Vokalpolyphonie [23] beschreibt der dänische Komponist und Musikwissenschaftler Methoden zur „Komposition“ von kontrapunktischen Sätzen in Bezug auf den Stil Giovanni Pierluigi Palestrinas (ca. 1525–1594). Jeppesen gliedert das kontrapunktische Material in rhythmische Varianten - den *Gattungen* - sowie Sätze unterschiedlicher Stimmenzahl und gibt für jede dieser Varianten präzise Rahmenbedingungen und Regeln zu deren stilkonformer Generierung an. Einstimmige Melodien der ersten Gattung beschreiben hier Notenfolgen gleicher rhythmischer Dauer, die als sog. *Cantus Firmi* die Grundlage eines mehrstimmigen Satzes bilden können. Da melodisches Material dieser Gattung durch die Regeln Jeppesens im Hinblick auf seine Stilkonformität gut überprüft werden kann, ermöglichen sich dadurch einigermaßen objektive Bewertungen von Generierungen dieses nichtwissensbasierten Systems. Abb. 4 zeigt ein Beispiel für eine Durchschreitung eines Korpus dorischer Melodien, wobei die hellgrauen Bereiche die erfassten Kontexttiefen, die dunkleren Bereiche die Spur der „Context Snake“ verdeutlichen – das Resultat ist in den letzten beiden Zeilen ersichtlich.

⁷ Alberto de Campo implementierte eine Version des Algorithmus in Super Collider als eine Klasse, die über die symbolische Verarbeitung von Zeichenketten hinaus eine Reihe von interessanten Zusatzfunktionen bereitstellt. (Diese Implementation wird als extension in absehbarer Zeit allgemein zum download zur Verfügung stehen)

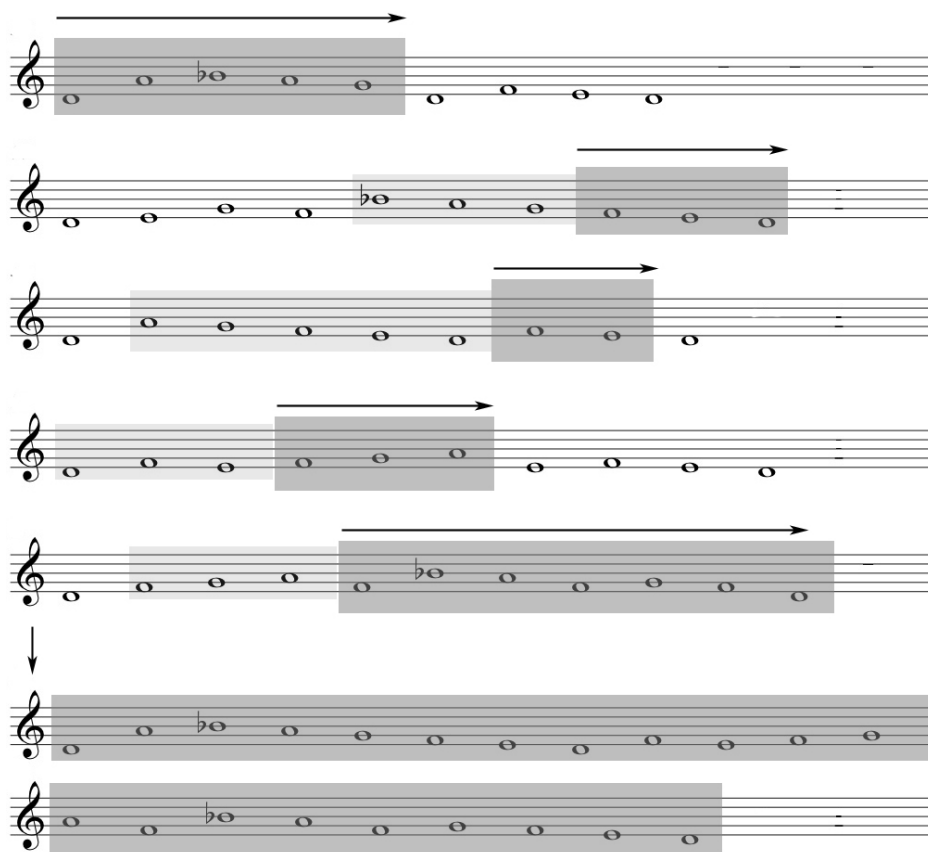


Abb. 3: Generatives Durchschreiten eines Korpus dorischer Chormelodien.

Die, vom Algorithmus generierten Cantus Firmi entsprachen in vielen Beispielen den stilistischen Vorgaben und Regeln Jeppesens – eine Evaluierung erfolgte unter Mitwirkung von Klaus Lang (Prof. für kirchliche Komposition an der KUG).

Weitere interessante Ergebnisse ließen sich auch bei der Generierung von rhythmisierten Stimmen aus Palästrina Messen erreichen, die Resultate wurden aber noch keiner eingehenderen Prüfung unterzogen

Generierung von Kernmelodien

Wahl des Korpus

Um auch im Hinblick auf die Evaluierung stilistisch einheitliches Material zu generieren wurde durch Rainer Schuetz ein Korpus von 87 Kernmelodien zur Verfügung gestellt.

Es handelt sich bei diesem Repertoire um pélog-Kernmelodien mit Gongperioden, bestehend aus jeweils 32 metrischen Einheiten (acht gatrás zu je vier „Schlägen“) aus dem Stil „Ladrang“.

Generierung und Klassifizierung von Gongperioden und Einleitungssphrasen

In einem ersten Schritt wurden durch den Generierungsalgorithmus Gongperioden und Einleitungssphrasen erzeugt, eine Auswahl erfolgte im Hinblick auf eine – den Beispielen des Korpus entsprechende – Dauernstruktur⁸.

Beispiele für generierte Gongperioden in Buchstaben- und Zahlennotation:

-FGI -FGI) -FGI^ -GFE) -FGI^ -GFE) II-F^ GIJI@
-6.7.2 -6.7.2) -6.7.2^ -7.6.5.) -6.7.2^ -7.6.5.) 22-6.^ 7.232@

-G-F -G-M) -P-N^ -J-I) -M-N^ -J-I) -J-I^ -G-F@
-7.-6. -7.-6) -2'-7^ -3-2) -6-7^ -3-2) -3-2^ -7.-6.@

JIJL NMJI) JIJL^ NMJI) JL--^ NMJI) HHJI^ -GFE@
3235 7632) 3235^ 7632) 35--^ 7632) 1132^ -7.6.5.@

-NNN MLMN) -NML^ JLMN) -NN^ -NNMN) PP-N^ MLJL@
-777 6567) -765^ 3567) -77^ 7767) 2'2'-7^ 6535@

II-- IJIG) JIGF^ IJIG) LL--^ IJLM) LLML^ JJLM@
22-- 2327.) 327.6.^ 2327.) 55--^ 2356) 5565^ 3356@

NN-- NMLM) JLMN^ MLJI) MLMN^ MLJI) GGJI^ -GEF@
77-- 7656) 3567^ 6532) 6567^ 6532) 7.7.32^ -7.5.6.@

Generierung von Kernmelodien

Da die Größe des aktuellen Korpus für eine Generierung gesamter Kernmelodien nur einen eingeschränkten Spielraum ermöglicht wurden Kernmelodien durch Kombination generierter Gongperioden erzeugt. Aus Beobachtungen des Korpus wurden einfache Klassifizierungsfunktionen⁹ erstellt, die in weiterer Folge zur Kombination der Gongperioden verwendet wurden.

Beispiele für generierte Kernmelodien in Buchstaben- und Zahlennotation:

buka
-GFG IJIG FGFE CEFG@
-7.6.7. 2327. 6.7.6.5. 3.5.6.7.@
-I-G -J-I) --FG^ -J-I) -GFG^ IJIG) FGFE^ CEFG@
-2-7. -3-2) --6.7.^ -3-2) -7.6.7.^ 2327.) 6.7.6.5.^ 3.5.6.7.@
--JL MNLM) JLMN^ MLIJ) LMLJ^ IGFG) -GFE^ CEFG
--35 6756) 3567^ 6523) 5653^ 27.6.7.) -7.6.5.^ 3.5.6.7.

⁸ Durch die Funktionsweise des Algorithmus können natürlich auch Gongperioden generiert werden, die einer 32 fachen Unterteilung nicht entsprechen.

⁹ Z.B. Anzahl gleicher Tonhöhen, hohe, tiefe, steigende und fallende Tonhöhenstrukturen etc.

buka
 -JLM NMLJ IGIJ IGEF@
 -356 7653 27.23 27.5.6.@
 -EFG -J-I) --FG^ -J-I) --FG^ I-JI) KJKJ^ IGEF@
 -5.6.7. -3-2) --6.7.^ -3-2) --6.7.^ 2-32) 4343^ 27.5.6.@
 CC-E FGFE) -EEE^ CEFG) -GIJ^ KJIG) FGFE^ JLMN@
 3.3.-5. 6.7.6.5.) -5.5.5.^ 3.5.6.7.) -7.23^ 4327.) 6.7.6.5.^ 3567@
 NN-- NMLM) NNMN^ MLIJ) LMLJ^ IGEF) IGIJ^ IGEF@
 77-- 7656) 7767^ 6523) 5653^ 27.5.6.) 27.23^ 27.5.6.@

buka
 ---G JIFG IJIG FGFE CEFG@
 ---7. 326.7. 2327. 6.7.6.5. 3.5.6.7.@
 II-- IJI) GGIJ^ MLJI) --FG^ IJIG) FGFE^ CEFG@
 22-- 2232) 7.7.23^ 6532) --6.7.^ 2327.) 6.7.6.5.^ 3.5.6.7.@
 -I-G -J-I) --FG^ -J-I) -GFG^ IJIG) FGFE^ CEFG@
 -2-7. -3-2) --6.7.^ -3-2) -7.6.7.^ 2327.) 6.7.6.5.^ 3.5.6.7.@

Ausblick

Im Hinblick auf eine umfassendere Auseinandersetzung mit dieser Thematik wären folgende weitere Aufgabenstellungen von Interesse

- Evaluierung der Resultate verschiedener minimal notwendiger Kontexttiefen.
- Evaluierung der Resultate durch professionelle Musiker des Gamelan.
- Umfassende Erschließung von Gesetzmäßigkeiten in Bezug auf Möglichkeiten der Kombination von Einleitungssphrasen und Gongperioden aus einem umfangreichen Korpus.
- Generierung von Kernmelodien verschiedener Stilrichtungen unter zugrunde Legung eines umfangreichen Korpus.
- Klassifizierungsalgorithmen für beliebige Kernmelodien im Hinblick auf deren Zugehörigkeit zu einem bestimmten Stil in Analogie zur Erkennung formaler Sprachen durch Automatenmodelle.
- Ableitung eines umfangreichen Regelsystems durch die Zusammensetzung evaluierter Generierungen.
- Anwendung der beschriebenen Analyse- und Generierungsmodelle im Hinblick auf musikalische Strukturen unterschiedlicher Provenienz.

Literatur

- [1] Allen M (2002) Harmonising chorales in the style of Johann Sebastian Bach. Thesis, University of Edinburgh, 2002
- [2] Becker A, Becker JO (1979) A Grammar of the musical genre Srepegan. In: Journal of Music Theory, 23, 1979
- [3] Bel B (1998) Migrating musical concepts – an overview of the Bol Processor. In: Computer Music Journal, 22/2, 1998
- [4] Bel B, Kippen J (1992) Bol Processor Grammars. In: Balaban M, Ebcioglu K, Laske O (eds) Understanding music with AI. The AAAI Press, California
- [5] Bel B, Kippen Jim (1989) The identification and modelling of a percussion “language”, and the emergence of musical concepts in a machine-learning experimental setup. In: Computers and the Humanities, 23/3, 1989
- [6] Bellgrad MI, Tsang CP (1994) Harmonizing music the Boltzmann way. In: Connection Science, 6, 1994
- [7] Biles JA (1994) GenJam: a genetic algorithm for generating jazz solos. In: Proceedings of the 1994 International Computer Music Conference, ICMC, San Francisco
- [8] Blacking J (1982) What languages do musical grammars describe? In: Baroni M, Callegari L (eds) Musical Grammars and Computer Analysis. Casa Editrice Leo S. Olschki, Florence
- [9] Brooks FP, Hopkins AL, Neumann PG, Wright WV (1957) An experiment in musical composition. In: IRE Transactions on Electronic Computers, 1957. In: Schwanauer SM, Levitt DA (eds) (1993) Machine models of music. The MIT Press, Massachusetts
- [10] Burton AR (1998) A hybrid neuro-genetic pattern evolution system applied to musical composition. Dissertation, University of Surrey, 1998
- [11] Chemellier M (2001) Improvising jazz chord sequences by means of formal grammars.<http://recherche.ircam.fr/equipes/repmus/marc/publi/jim2001/icmc2001.pdf>
Cited 27 Aug 2007
- [12] Cope D (1987) An expert system for computer-assisted composition. In: Computer Music Journal 11/4
- [13] Cope D (1991) Computers and musical style. Oxford University Press, Oxford

- [14] Cope D (1996) Experiments in Musical Intelligence. AR-Editions, Wisconsin
- [15] Cope D (2000) The algorithmic composer. AR-Editions, Wisconsin
- [16] Cope D (2001) Virtual music: computer synthesis of musical style
- [17] Dodge C, Jerse TA (1997) Computer Music: synthesis, composition, and performance, 2nd edn. Schirmer Books, New York, p. 364-368
- [18] Drummond, Barry (2006) Gendhing Jawa – Jawanese Gamelan Notation. <http://muse.calarts.edu/~drummond/gendhing.html>. Zugriff: 12. Juni 2006
- [19] Ebcioglu K (1992) An expert system for harmonizing four-part chorales. In: Balaban M, Ebcioglu K, Laske O (eds.) Understanding music with AI. The AAAI Press, California
- [20] Farbood M, Schoner B (2001) Analysis and synthesis of Palestrina-style counterpoint using Markov chains. In: Proceedings of International Computer Music Conference. Havana, Cuba, 2001
- [21] Hild H, Feulner J, Menzel W (1992) HARMONET: A neural net for harmonizing chorales in the style of J.S.Bach. In: Advances in Neural Information Processing Systems 4 (NIPS*4), San Mateo, CA, 1992
- [22] Hughes DW (1991) Grammars of on-Western music. In: Howell P, West R, Cross I (eds) Representing musical structure. Academic Press, London
- [23] Jeppesen, Knud (1985) Kontrapunkt. Lehrbuch der klassischen Vokalpolyphonie. Breitkopf U. Haertl
- [24] Kohonen T (1987) Self-learning inference rules by dynamically expanding context. In: Proceedings of the IEEE First Annual International Conference on Neural Networks, San Diego, 1987
- [25] Kohonen T (1989) A self-learning musical grammar, or “Associative memory of the second kind”. In: Proceedings of the International Joint Conference on Neural Networks, New York, 1989
- [26] McIntyre RA (1994) Bach in a Box: the evolution of four part Baroque harmony using the genetic algorithm. In: Proceedings of the First IEEE Conference on Evolutionary Computation, Orlando, Florida, 1994
- [27] Mozer MC (1994) Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multiscale processing. In: Connection Science, 1994
- [28] Nevill-Manning CG, Witten IH (1997) Identifying hierarchical structure in sequences: A linear-time algorithm. In: Journal of Artificial Intelligence Research, 7, 1997

- [29] Pachet F (1999) Surprising Harmonies. In: International Journal of Computing Anticipatory Systems, 1999
- [30] Papadopoulos G, Wiggins G (1998) A genetic algorithm for the generation of jazz melodies. In: STeP98, Jyväskylä, Finland, 1998
- [31] Pelinski R (1982) A generative grammar of personal Eskimo songs. In: Baroni M, Callegari L (eds) Musical grammars and computer analysis. Casa Editrice Leo S. Olschki, Florence.
- [32] Phon-Amnuaisuk S, Tuson A, Wiggins G (1999) Evolving musical harmonisation. In: Proceedings of the International Conference on Adaptive and Natural Computing Algorithms, Porto Roz, Slovenia, 1999
- [33] Phon-Amnuaisuk S, Wiggins GA (1999) The four-part harmonisation problem: a comparison between genetic algorithms and a rule-based system. In: Proceedings of the AISB'99 Symposium on Musical Creativity
- [34] Phon-Amnuaisuk S, Snail A, Wiggins G (2002) A computational model for chorale harmonisation in the style of J.S. Bach. In: Proceedings of ICMPC7 (the 7th International Conference on Music Perception and Cognition), Sydney, Australia, 2002
- [35] Pickvance, Richard (2005) A Gamelan Manual: a player's guide to the central Javanese gamelan. Jaman Mas Books, London
- [36] Ribeiro P, Pereira FC, Ferrand M, Cardoso A (2001) Case-based melody generation with MuzaCazUza. In: Proceedings of the AISB'01 Symposium on Artificial Intelligence and Creativity in Arts and Science, 2001
- [37] Schwanauer SM (1993) A learning machine for tonal composition. In: Schwanauer SM, Levitt DA (eds.) (1993) machine models of music. The MIT Press, Massachusetts
- [38] Steedman M (1984) A generative grammar for jazz chord sequences. In: Music Perception 2, 1984
- [39] Steedman M (2003) Formal grammars for computational musical analysis. In: INFORMS Atlanta October 2003
- [40] Sundberg J, Lindblom B (1991) Generative theories for describing musical structure. In: Howell P, West R, Cross I (eds) (1991) Representing musical structure. Academic Press, London
- [41] Todd PM (1989) A connectionist approach to algorithmic composition. In: Computer Music Journal, 13/4, 1989
- [42] Widmer G (1992) Qualitative perception modeling and intelligent musical learning. In: Computer Music Journal 16/2, 1992

[43] Widmer G (1992) The importance of basic musical knowledge. A knowledge intensive approach to machine learning. In: Balaban M, Ebcioglu K, Laske O (eds.) (1992) Understanding music with AI. The AAAI Press, California

[44] Ziv J, Lempel A (1978) Compression of individual sequences via variable-rate coding. In: IEEE Transactions On Information Theory, 24/5, September, 1978

Siehe auch weitere Abschlussberichte im Rahmen des „Virtual Gamelan Graz (VGG)“:

Alois Sontacchi, Franz Zotter, Gerhard Eckel, Robert Höldrich:
IEM Report 42/07: Klangmodellierung - ein Beitrag zur musikalischen Akustik
Endbericht zum Projekt „Virtual Gamelan Graz (VGG)“

Gerd Grupe, Rainer Schütz, „Abschlussbericht zum Projekt „Virtual Gamelan Graz (VGG)“
IME VGG Endbericht, 2007.