AKADEMIA MUZYCZNA w KRAKOWIE
WYDZIAŁ TWÓRCZOŚCI,
INTERPRETACJI I EDUKACJI
MUZYCZNEJ

Utwór orkiestrowy pt. *Rozbłyski* jako przykład zastosowania CAC w moim języku kompozytorskim

Praca doktorska

mgr Piotr Komorowski
Promotor: dr hab. Sławomir Czarnecki

KRAKÓW 2010

Autor chciałby wyrazić swoje podziękowania następującym osobom:

- Promotorowi dr hab. Sławomirowi Czarneckiemu, za pomoc w trakcie pisania niniejszej pracy
- Dyrektor Filharmonii Pomorskiej w Bydgoszczy pani Eleonorze Harendarskiej
- Dyrygentowi Wojciechowi Rodkowi oraz muzykom Filharmonii Pomorskiej w Bydgoszczy za przygotowanie prawykonania utworu "Rozbłyski" oraz wyrażenie zgody na jego rejestrację
- Wszystkim twórcom wolnego oprogramowania komputerowego, bez którego powstanie tej pracy byłoby niemożliwe. Wśród nich są:
  - Carlos Agon, Gérard Assayag - autorzy programu *OpenMusic*, którzy rezultat swojej nadzwyczajnej pracy zdecydowali się udostępnić wszystkim, publikując kod źródłowy programu na licencji GPL
  - Karim Haddad, który jest jednym z twórców dokumentacji *OpenMusic*, napisał wykorzystywaną przy tworzeniu przykładów nutowych w tej pracy bibliotekę *OMLILY*, a także dopomógł autorowi w jej uruchomieniu
  - Jean Bresson, współtwórca programu *OpenMusic*, który udzielił autorowi niniejszej pracy wskazówek niezbędnych do jego skompilowania
  - Michael Klingbeil, autor doskonałego i udostępnionego nieodpłatnie programu do analizy, edycji i syntezy spektralnej *SPEAR*

Spis treści

Spis rysunków

Rozdział 1

Wstęp

CAC, czyli *komputerowo wspomagana kompozycja* stała się w ciągu ostatnich kilkunastu lat standardowym narzędziem w pracy wielu wybitnych twórców muzyki. Sytuacja dzisiaj jest przy tym zupełnie inna niż w początkach istnienia tej dziedziny, czyli w połowie XX wieku. Kompozytorów wykorzystujących w swej pracy komputer, w odróżnieniu od pionierów muzyki komputerowej, nie traktuje się już z powątpiewaniem kwitowanym wzruszeniem ramion. Powodów zmiany stosunku do CAC jest wiele. Przede wszystkim trudno dziś wyobrazić sobie życie milionów ludzi cywilizacji zachodniej bez komputera. Ten cichy i usłużny, choć czasem potrafiący przysporzyć poważnych problemów towarzysz człowieka, zawłaszcza coraz więcej obszarów naszego życia. Czemuż więc nie miałby wkroczyć także do dziedziny twórczości artystycznej? Wydaje się to być logicznym następstwem rozwoju technologii, stanowiącym kolejny epizod w długim ciągu przemian prowadzących od glinianej tabliczki do palmtopa. Sytuacja nie jest jednakże oczywista i jednowymiarowa. Z jednej strony komputer przyśpiesza, ułatwia lub uprzyjemnia życie dzisiejszego człowieka, z drugiej strony jednak czyni je także coraz bardziej skomplikowanym, stawiając przed nami coraz to wyższe wymagania, żadając opanowania coraz to nowych umiejętności i dostosowywania się do coraz szybszego tempa przemian technologicznych. Informatyzacja zachodnich społeczeństw jest faktem, i jak zawsze gdy dochodzi do zmian, jedni traktują tę sytuację jak błogosławieństwo, inni jak przekleństwo. Wydaje się,

że niezauważanie tego zjawiska lub próba przeciwstawienia się mu jest dziś tak samo nieproduktywna, jak bezowocna była dwieście lat temu walka luddystów przeciw rewolucji przemysłowej, wznoszona pod hasłami niszczenia maszyn fabrycznych. Czarnowidztwo, czy też nadmierny optymizm nigdy nie przynoszą właściwych rozwiązań.

Zagadnienie będące głównym tematem niniejszej pracy jest stosunkowo bogato reprezentowane w literaturze obcej (głównie angielsko- i francuskojęzycznej), a mimo tego nie doczekało się jeszcze omówienia w polskim piśmiennictwie muzykologicznym. Brak pozycji książkowych, czy choćby artykułów prasowych przybliżających ten problem polskiemu czytelnikowi. Doskonała skądinąd *Muzyka elektroniczna* autorstwa Włodzimierza Kotońskiego [60], mimo że w wersji uzupełnionej wydana została w 2002 roku, nie zawiera żadnych informacji na temat współcześnie wykorzystywanych systemów CAC. Wobec tego, głównym źródłem informacji wykorzystanych podczas komponowania utworu pt. *Rozbłyski* oraz pisania niniejszej pracy była literatura obca, głównie angielskojęzyczna, oraz w niewielkim stopniu francuskojęzyczna. Pierwszym i nieocenionym źródłem wiedzy była dokumentacja programów komputerowych, opublikowane w internecie prace naukowe oraz artykuły w czasopismach fachowych. Należy jednakże zaznaczyć, że praca ta jest jedynie opisem zastosowania *komputerowo wspomaganej kompozycji* przez jednego kompozytora i to na przykładzie jednego utworu, a wobec tego autor nie rości sobie pretensji do całościowego i wyczerpującego omówienia tego niezmiernie złożonego zagadnienia.

Skala zastosowań komputerów w kompozycji jest bowiem niezwykle szeroka: od roli swoistego kalkulatora, aż do "kompozytora". Na dwóch krańcach znajdują się: z jednej strony systemy komputerowo wspomaganej kompozycji algorytmicznej (CAAC), potrafiące na podstawie dostarczonych danych skomponować w całości utwór muzyczny, z drugiej zaś swego rodzaju "asystenci" kompozytora, wykonujący jedynie określone zadania, najczęściej żmudne i mechaniczne czynności. Trzeba jednak stwierdzić, że zakres zastosowania CAC zależy w każdym przypadku od postawionych przez twórcę celów i wytyczonych przez niego granic. Założeniem przyjętym przez autora niniejszej pracy było przede wszystkim wykorzystanie komputera w różnego

rodzaju obliczeniach, generowaniu i modyfikowaniu materiału dźwiękowego, analizowaniu zjawisk o skali tak małej, że nieuchwytnej zmysłami, uzyskaniu możliwości szybkiego zweryfikowania pomysłów kompozytorskich. Reszta, czyli przede wszystkim praca koncepcyjna, wybór materiału, projektowanie całości, czy ostateczna weryfikacja rezultatów była zadaniem kompozytora.

Niniejsza praca przynosi opis cząstki możliwości wykorzystania systemów komputerowych w kompozycji na przykładzie utworu orkiestrowego pt. *Rozbłyski*. Ta siedemnastominutowa kompozycja napisana została w 2009 roku. Tytuł zdradza rodzaj inspiracji, która odwołuje się do zjawisk natury wizualnej. Muzyka *Rozbłysków*, rodząca się z samej natury dźwięku, przepływająca nakładającymi się i odbijającymi od siebie rytmicznymi falami, stara się przełożyć wrażenia wizualne, grę świateł i kolorów na język dźwięków, brzmień, harmonii i rytmu. Jej koncepcja oparta zatem została na próbie poszukiwania analogii między tymi dwoma światami. Język dźwiękowy tej kompozycji charakteryzuje wykorzystanie zjawisk akustycznych, fizycznych właściwości fal dźwiękowych, w czym nieocenioną rolę odegrały narzędzia informatyczne, przede wszystkim zaś najbardziej dziś rozwinięte, najnowocześniejsze środowisko CAC, czyli stworzony w paryskim IRCAM-ie program *OpenMusic*.

W rozdziale drugim przedstawione zostaną teoretyczne, historyczne i technologiczne uwarunkowania komputerowo wspomaganej kompozycji. Na początek przeprowadzony zostanie ogólny podział tzw. muzyki komputerowej oraz zaproponowana zostanie odpowiednia terminologia. Następnie opisane zostaną podstawowe elementy programowania komputerów: algorytmy oraz języki programowania. Dalej w postaci krótkiego szkicu historycznego ukazana zostanie ewolucja myślenia algorytmicznego w muzyce i to zarówno w czasach historycznych, jak i w dobie dzisiejszej, czyli erze komputerów. W kolejnym podrozdziale omówione zostaną najważniejsze z punktu widzenia rozwoju CAC programy oraz systemy komputerowe służące kompozycji. Rozdział zamyka skrótowy opis narzędzi informatycznych wykorzystanych w pracy nad utworem pt. *Rozbłyski*.

Kolejne części pracy przynoszą konkretne przykłady zastosowań CAC. Ponieważ jednak omówienie określonych narzędzi informatycznych musi jednocześnie odwoływać się do przykładów muzycznych, a równoczesny opis

programów oraz generowanego przez nie materiału (włącznie z wyjaśnieniem rządzących nim zasad) nie sprzyjałby logice oraz przejrzystości wywodu, materiał ten prezentowany jest w dwóch kolejnych rozdziałach. Są one jednak komplementarne, tak że właściwy ich sens ujawnia się dopiero po prześledzeniu istniejących między nimi powiązań, które każdorazowo są w tekście sygnalizowane przy pomocy odpowiednich odwołań. Wobec tego w rozdziale trzecim przedstawione zostaną informacje niezbędne dla zrozumienia głównej idei utworu, tzn. obsada instrumentalna oraz budowa formalna wraz z zasadami kształtowania formalnego. Główną część tego rozdziału stanowi natomiast opis organizacji materiału dźwiękowego w aspekcie wysokościowym, jako ten element języka kompozytorskiego, w którym zastosowanie komputerowo wspomaganej kompozycji miało znaczenie pierwszoplanowe i przez to determinujące ostateczny kształt utworu. W rozdziale czwartym znajdują się natomiast kody programów, użytych podczas komponowania utworu pt. *Rozbłyski*. Ponieważ *OpenMusic* jest wizualnym językiem programowania kody przedstawione są w postaci tzw. łat (ang. patch), stanowiących graficzną reprezentację programu. W niektórych, wymagających tego miejscach, znalazły się także objaśnienia dotyczące zasady funkcjonowania danego programu.

Koncepcja tej pracy zakłada przede wszystkim ukazanie w jaki sposób zastosowanie komputerowo wspomaganej kompozycji może pomóc współczesnemu kompozytorowi w pracy twórczej, stając się elementem jego techniki oraz ważnym czynnikiem kształtującym język dźwiękowy. Praca ta w możliwie zwięzły sposób próbuje przedstawić pewien zakres możliwości CAC na konkretnym przykładzie dzieła artystycznego. Z drugiej strony materiał tu zawarty może także służyć uzupełnieniu luki istniejącej w polskim piśmiennictwie dotyczącym zarówno historii, współczesności, jak i zastosowań komputerowo wspomaganej kompozycji.

Rozdział 2

Problematyka CAC, czyli komputerowo wspomaganej kompozycji

2.1 Zagadnienia ogólne

Muzyka komputerowa rozwija się ogólnie rzecz biorąc w dwu podstawowych odnogach, których kierunki symbolicznie wyznaczają dwa historyczne wydarzenia z 1957 roku. Max Mathews stworzył wówczas program *Music I*, będący w rzeczywistości pierwszym ukierunkowanym muzycznie językiem programowania przeznaczonym do cyfrowej syntezy dźwięku, a Lejaren Hiller (wraz z Leonardem Isaacsonem) pierwszą oryginalną kompozycję stworzoną przy użyciu komputera – *Iliac Suite for String Quartet*. Miller Puckette, jeden z najważniejszych twórców muzycznego oprogramowania, autor takich programów jak: *Max* (nazwa jest hołdem dla Maxa Mathewsa) i *Pure Data*, zaproponował w związku z tym następującą terminologię:

- *Computer Generated Music* – dla muzyki syntezowanej przy użyciu komputera, oraz
- *Computer Aided Composition* – dla muzyki, której proces kompozycyjny oparty jest na komputerowym manipulowaniu symbolicznymi

przedstawieniami dźwięku [13, s. ix]

CAC to akronim angielskich słów Computer Aided Composition (niekiedy również tłumaczone jako Computer Assisted Composition), oznaczających *komputerowo wspomaganą kompozycję*. W języku francuskim istnieje skrót CAO, czyli Composistion Assisté par Ordinateur. Systemy CAC skupiają się w głównej mierze na formalnej strukturze muzyki i ten aspekt odróżnia je od oprogramowania służacego do syntezy dźwięku, tj.: *Chuck*, *SuperCollider*, czy *CSound*. Do najbardziej znanych, współcześnie wykorzystywanych programów CAC należą: *Common Music*, *PatchWork* oraz dwaj jego spadkobiercy: *PWGL* i *OpenMusic*. W języku polskim brak jeszcze jakiegoś utrwalonego i powszechnie używanego określenia oznaczającego oprogramowanie służące komputerowo wspomaganej kompozycji. Ponieważ termin *oprogramowanie muzyczne* jest zbyt ogólnikowy i nie oddaje w pełni znaczenia oryginału angielskiego, dla potrzeb niniejszej pracy przyjęto termin *oprogramowanie CAC*. Odpowiednikiem CAC w innej dziedzinie działalności jest powszechnie stosowany również w języku polskim termin CAD – Computer Aided Design, czyli komputerowo wspomagane projektowanie[1].

Również w języku angielskim zdaje się obecnie panować rodzaj terminologicznej dowolności, skądinąd zrozumiałej w przypadku stosunkowo młodej gałęzi wiedzy. Oprócz często stosowanych, lecz nieprecyzyjnych definicji tj.: *algorithmic composition, automatic composition, composition pre-processing, computer-aided composition, computer composing, procedural composition, score synthesis*, niektórzy amerykańscy autorzy (Martin Supper, Christopher Ariza) używają innego jeszcze terminu-hybrydy: Computer-Aided Algorithmic Composition (CAAC), oznaczający komputerowo wspomaganą kompozycję algorytmiczną [20]. Ponieważ jednak stosowanie programów komputerowych w kompozycji nie zawsze wiąże się z wyznaczaniem kształtu kompozycji przy użyciu ścisłych reguł algorytmicznych, autor niniejszej pracy przyjmuję ten termin jako szczególny przypadek oprogramowania CAC.

[1]CAD jest jednym z narzędzi używanych przez inżynierów, architektów i projektantów i znajduje zastosowanie m.in. w tworzeniu cyfrowych makiet wyrobu, wykonywaniu dokumentacji rysunkowej, opracowywaniu i zarządzaniu bazami danych dotyczących np. własności materiałowych, a także symulacji, wizualizacji i animacji [5].

Dalsza część rozdziału poświęcona będzie rozwojowi oraz zastosowaniu różnych procedur algorytmicznych służących tworzeniu muzyki, a także rozwojowi komputerowo wspomaganej kompozycji. W pracy tej przyjęto perspektywę, w której za podstawę klasyfikacji programów komputerowych służących kompozycji muzycznej przyjęto stopień możliwej ingerencji kompozytora w przebieg i realizację programu. Wobec tego założenia oprogramowanie CAC, obejmujące swoją funkcjonalnością szeroką gamę zastosowań, można podzielić na dwa rodzaje:

- programy do kompozycji automatycznej, zwanej również algorytmiczną (zakres znaczeniowy pokrywa się w tym przypadku z teminem CAAC) oraz
- programy asystujące, inaczej mówiąc systemy konwersacyjne (ang. *interactive systems*)

2.2 Uwarunkowania CAC

> "W zasadzie komputery mogą być zaprogramowane tak, aby wykonać prawie każde wyobrażalne zadanie lub rozwiązać prawie każdy wyobrażalny problem pod warunkiem, że metoda jego rozwiązania może być jasno zdefiniowana, a dane w jasny sposób przedstawione[2]."
>
> Edaurdo Reck Miranda, *Composing Music with Computers*[69, s. 42]

Zdanie to streszcza dzisiejszą wiedzę na temat programowania komputerów. Dziedzina ta, oprócz uwarunkowań czysto technicznych, sprzętowych (hardware) podlega również czynnikom natury intelektualnej (software). Zacytowana wypowiedź wskazuje także na dwa istotne w programowaniu komputerów elementy: (a) dokładny przepis realizacji zadania, czyli algorytm oraz (b) język, w którym zostanie on wyrażony. Poniżej przedstawiony zostanie opis tych zagadnień, ze szczególnym uwzględnieniem ich zastosowania w dziedzinie komputerowo wspomaganej kompozycji.

[2]Tłumaczenie wszystkich zamieszczonych w pracy cytatów pochodzi od autora.

2.2.1 Algorytmy

Pojęcie *algorytm* wywodzi się od zlatynizowanej formy imienia słynnego perskiego matematyka Muhammada ibn Musa al-Chorezmi. Około roku 820 n.e. napisał on traktat, w którym przedstawił m.in. nowy, dziesiętny system liczbowy wynaleziony cztery wieki wcześniej w Indiach oraz metodę przeprowadzania obliczeń arytmetycznych opartą na określonych, powtarzających się operacjach. Łacińskie tłumaczenie tego dzieła ukazało się około roku 1120 jako *Algorismi de numero Indorum*. Jego najrozmaitsze odpisy, z których najstarszy pochodzi z roku 1143 rozpoczynały się od słów *Dixit algorizmi* [56, s. 82-87]. W ten sposób narodził się termin *algorytm*, który w *Słowniku języka polskiego* zdefiniowany jest jako:

> "(...)dokładny przepis wykonania w określonym porządku skończonej liczby operacji, pozwalający na rozwiązanie każdego zadania danego typu (...) [100]"

Idea formalizacji języka dźwiękowego i wykorzystania procedur algorytmicznych w kompozycji jest dużo starsza od techniki komputerowej, a elementów myślenia algorytmicznego można doszukać się w technikach kompozytorskich stosowanych na długo przed wynalezieniem maszyn liczących. W kolejnym podrozdziale znajdziemy kilka przykładów.

2.2.2 Języki programowania

Rozwój języków programowania stał się jednym z najważniejszych czynników w rozwoju komputerowo wspomaganej kompozycji. Jednocześnie wybór konkretnego języka posiada duże znaczeniu dla rozwiązania określonego problemu, gdyż wpływa zarówno na sposób formalizacji idei kompozytorskich, jak i na przebieg ich realizacji. Dzieje się tak dlatego, że użycie danego języka sugeruje pewien sposób wyrażania, który nie byłby porządany lub możliwy w innym języku [21, s. 2].

Z uwagi na ogromną liczbę wykorzystywanych dzisiaj języków programowania, nie jest łatwo przeprowadzić ich jednoznaczną klasyfikację. W podziale

historycznym stosuje się kategorię generacji[3]. Do pierwszej generacji języków programowania zaliczamy tzw. *języki maszynowe*, czyli jedyne jakie w rzeczywistości "rozumiane" są przez komputer. Każdy rozkaz reprezentowany jest w nich przez ciąg bitów zakodowanych zgodnie z formatem określonym przez listę rozkazów mikroprocesora. W praktyce posługiwanie się kodem maszynowym, zwanym również *językiem wewnętrznym*, jest niezwykle czasochłonne i w przypadku dużych programów narażone na błędy [61, s. 12]. Języki programowania drugiej generacji to *asembler* oraz jego rozwinięcię w postaci *makroasemblera*, posługujące się notacją symboliczną, w której elementy rozkazu, tj. rodzaj operacji, tryb adresowania oraz nazwa argumentu, wyrażone są za pomocą odpowiednio dobranych słów, skrótów, umownych znaków i liczb dziesiętnych. Asembler, a także makroasembler nazywane są czasami *językami symbolicznymi*. Są one bardzo odległe od języków naturalnych, a ponieważ są ściśle powiązane z językiem wewnętrznym, programy pisane w nich dla jednego procesora nie mogą być wykonywane przez inne. Obydwa te problemy rozwiązane zostały w trzeciej generacji, czyli w tzw. *językach wysokiego poziomu* (Fortran, Lisp, Basic, C), które umożliwiają niezależny sprzętowo zapis programu. Pojęcia techniczne zastąpiono w nich konstrukcjami i notacją zaczerpniętą z matematyki, a słowami kluczowymi oznaczającymi instrukcje i typy argumentów są słowa i zwroty języka angielskiego. *Ukierunkowane języki programowania* stanowią czwartą geneację. Wśród nich przytoczyć możemy m.in. stworzone specjalnie dla potrzeb syntezy dźwięku języki z rodziny *Music* (Mathews, od 1957), czy wywodzący się z nich i posługujący podobną składnią Csound (Vercoe, od 1985). Inny przykład stanowią języki stworzone z myślą o symulacji procesów komponowania, jak *Project1* (Koenig, 1964), *ST* (Xenakis, 1963), czy systemy hybrydowe sterujące syntezą dźwięku przy pomocy urządzeń analogowych, jak *Groove* (Mathews, 1970), czy *Muzys* (Grogono, 1976) [60, s. 232-233]. Wzbogacają one koncepcję języków wysokiego poziomu o zastosowanie narzędzi (najczęściej w postaci podprogramów), ułatwiających rozwiązanie określonego typu problemów. Języki piątej generacji opierają się na programowaniu skierowanym na rozwiązywanie problemu w ramach wyznaczonych ram (ang. constraints

[3]Przedstawiony podział wg [79, s. 62-63]

programming) oraz programowaniu logicznym, dzięki czemu umożliwiają zapis problemu bez konieczności implementacji dokładnego przebiegu, czy algorytmu. Przykład stanowi wykorzystywany także w CAC *Prolog*, oparty na rachunku predykatów. Klasyfikacja języków programowania wg generacji nie oznacza jednocześnie układu hierarchicznego, w tym sensie, że najstarsze języki trzeciej generacji tj. Fortran, czy Lisp są nadal w użyciu, chociaż funkcjonują dziś w zmodyfikowanej formie.

Oprócz podziału historycznego istnieje także ujęcie tego zagadnienia według paradygmatów. *Języki imperatywne* (np. Fortran, Basic, Pascal) opierają się na przekazywaniu maszynie instrukcji, które wykonywane są jedna po drugiej. W *językach funkcyjnych*, tj. Lisp, które oparte są na rachunku lambda definiowane są funkcje, które mogą być wywołane niezależnie od ich miejsca w programie, którym może być także inna fukcja. *Języki obiektowe* (SmallTalk, Java) definiują obiekty reprezentujące autonomiczne jednostki danych i algorytmy. Obiekty posiadające taką samą strukturę wewnętrzną należą do tej samej *klasy*. Podstawowe cechy klasy mogą być hierarchiczne przenoszone do innych klas dzięki mechanizmowi *dziedziczenia*. Zachowanie obiektów jest natomiast opisywane przez *metody*. Oprócz wymienionych tutaj istnieje jeszcze wiele innych paradygmatów programowania, a dzisiejsze języki są najczęściej wieloparadygmatowe (przykładowo Common Lisp, będący rozwinięciem historycznego Lispa łączy progamowanie funkcyjne z obiektowym). Najmłodszą kategorię tworzą środowiska programowania wizualnego, w których obok rzeczywistego kodu manipulować można obiektami umieszczonymi na ekranie. Środowiska takie zbudowano dla języków różnych typów, a w dziedzinie muzyki komputerowej wyróżnić tu należy m.in.PatchWork, OpenMusic, Max, czy Pd[4].

Spośród wymienionych tu języków największą popularnością w komputerowo wspomaganej kompozycji cieszy się bez wątpienia Lisp. Należy on do najstarszych języków programowania i jest szczególnie przydatny do obliczeń symbolicznych, dzięki czemu znalazł szerokie zastosowanie w badaniach nad sztuczną inteligencją oraz CAC, w których to dziedzinach obliczenia

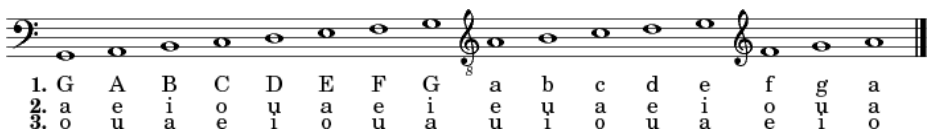
[4]Należy zaznaczyć, że na określenie tego typu pakietów programowych, w literaturze spotykamy różne terminy tj. “język”, “program”, “środowisko, czy “system”.

symboliczne są istotniejsze od obliczeń numerycznych [37, s. 193]. O jego popularności w dziedzinie CAC stanowi fakt, że na jego bazie stworzono m.in. najbardziej udane i najpopularniejsze współcześnie używane środowiska programistyczne wspomagające kompozycję: Common Music, Nyquist, PatchWork i PWGL oraz OpenMusic. W dalszej części rozdziału wielokrotnie odwoływać się będziemy do wymienionych tu języków.

2.3 CAC w zarysie historycznym

2.3.1 Kompozycja algorytmiczna w erze przedkomputerowej

Teoria muzyki niemal od początków swego istnienia wiązała tę dziedzinę sztuki przede wszystkim z matematyką, a stosowanie reguł i teorii matematycznych do komponowania muzyki posiada tradycję sięgającą wielu wieków wstecz. Wystarczy przywołać tu fundatora europejskiej myśli muzycznej Pitagorasa, który jako pierwszy powiązał dociekania akustyczne nad strojem muzycznym i obliczanie interwałów z proporcjami liczbowymi. Święty Augustyn, którego traktat *O muzyce* [108] wywarł duży wpływ na kształtowanie się teorii muzyki średniowiecznej, w swoich rozważaniach skupiał się między innymi na ukazywaniu związków rytmu muzycznego z obliczeniami arytmetycznymi. U zarania nowożytnych dziejów muzyki Guido z Arezzo stworzył nie tylko podstawy współczesnej notacji muzycznej, ale także pierwszy system automatycznej kompozycji muzycznej. W rozdziałach 15 i 17 traktatu *Micrologus de disciplina artis musicae* z około 1025 roku przedstawił metodę automatycznego tworzenia melodii do tekstów hymnów, wykorzystując do tego stworzoną przez siebie tabelę zawierającą samogłoski wraz z przypisanymi im wysokościami dźwięków (rysunek 2.1). Poszczególnym samogłoskom odpowiadają różne wysokości, a konkretne ukształtowanie melodii zależy od zastosowania reguł muzycznych opisywanych przez Guidona w innych rozdziałach traktatu.



Rysunek 2.1: Przypisanie samogłosek do wysokości dźwięków wg Guidona z Arezzo [79, s. 23].

Rozwój muzyki wielogłosowej prowadzi do powstania nowej dziedziny wiedzy kompozytorskiej, której nazwa pochodzi od pierwotnej techniki wykorzystywanej w *organum purum* – nota contra notam. Kontrapunkt rozumiany jako zbiór reguł i technik polifonicznych jest w istocie zestawem algorytmów, których realizacja umożliwia komponowanie muzyki wielogłosowej. Interesujący przykład zastosowania zapisu algorytmicznego w odniesieniu do realizacji utworu przez wykonawców stanowi natomiast kanon, którego rozkwit przypada na okres działalności kompozytorów tzw. *szkół franko-flamandzkich* (XV-XVI wiek). Kanon, będący zgodnie ze znaczeniem słowa (z gr.: prawidło, model, wzór) przepisem na stworzenie utworu polifonicznego z pojedynczej melodii, był początkowo notowany w postaci melodii oraz słownego komentarza (niekiedy w formie rebusu lub zagadki). W *Agnus Dei* z mszy *L'homme armé* Guillaume'a Dufay'a, kanon zapisany jest jako następująca zagadka:

"Cancer eat plenis et redeat medius",

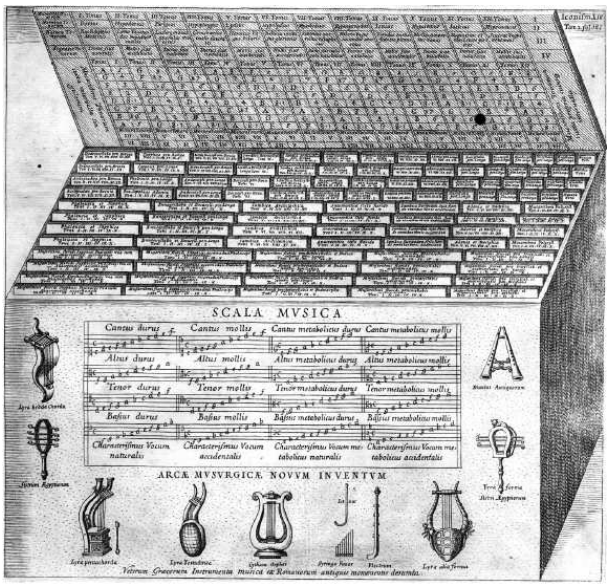
co można przetłumaczyć jako: "Niech rak przejdzie pełny (cały), a wróci średni". Znaczy to, iż cantus firmus powinien być początkowo śpiewany całymi wartościami (w ruchu wstecznym, gdyż jest to rak), a następnie zaśpiewany od tyłu w skróconych o połowę wartościach (tzn. w ruchu prostym, gdyż tym razem jest to retrogradacja raka). W przypadku-kanonów zagadek procedura algorytmiczna dotyczy zarówno realizacji utworu przez wykonawców, jak i procesu jego tworzenia, który regulowany jest przez reguły kontrapunktyczne – również algorytmiczne.

Idea komponowania automatycznego rozwinięta została następnie przez niemieckiego jezuitę i wszechstronnego naukowca Athanasiusa Kirchera. W pochodzącym z roku 1650 traktacie muzykologicznym pt. *Musurgia universalis sive ars magna consoni et dissoni* przedstawił on system kompozycji al-

gorytmicznej, umożliwiający automatyczne komponowanie czterogłosowych utworów kontrapunktycznych do tekstów łacińskich i to zarówno w stylu prostym, jak i ozdobnym. System ten, zwany *Arca Musarithmica*[5], składał się z trzech oznakowanych numerami i wartościami rytmicznymi kategorii, czyli tzw. syntagmatów (rysunek 2.2). Służyły one do łączenia czterocyfrowych kolum liczbowych z czterogłosowymi wzorami rytmicznymi. Kolumny te reprezentujące różne modusy, ułożone były w grupy od 2 do 12 jednostek, z których każda odpowiadała jednej sylabie umuzycznianego tekstu. W systemie Kirchera wysokości dźwięków reprezentowane były przy pomocy wartości liczbowych, co stanowi antycypację powszechnie używanych w teorii muzyki XX wieku klas wysokości (ang. *pitch class*). Każdy symbol klasy wysokości o określonym rozmiarze mógł być połączony z klasą rytmiczną tego samego rozmiaru, dając w rezultacie czterogłosowy utwór w stylu prostym. W stylu ozdobnym, w którym ilość głosów ulega zmianie, głosy łączone były tylko z wybranymi, odpowiadającymi im przebiegami rytmicznymi. Zastosowanie klas wysokości rozszerza liczbę możliwych kompozycji, gdyż na bazie tej samej abstrakcji umożliwia tworzenie odmiennych realizacji w różnych modusach [79, s. 24-26].

Modelem teoretycznym, na którym oparł swe idee Kircher, był projekt uniwersalnej maszyny logicznej – *Ars Magna*, opisywany choć niezrealizowany przez XIII-wiecznego filozofa pochodzącego z Majorki, Raimundusa Lullusa. Maszyna ta była próbą usystematyzowania wiedzy przy pomocy operacji matematycznych dokonywanych na twierdzeniach logicznych. Lullus był przekonany, że przy jej pomocy będzie w stanie odpowiedzieć na wszystkie pytania, nawet te dotyczące istoty Boga i dzięki temu będzie mógł nareszcie zakończyć spory religijne i doprowadzić do nawrócenia niewiernych. Uważał on również, że postęp w nauce dokonuje się dzięki tworzeniu nowych kombinacji skończonej liczby pojęć [56, s. 65]. Kircher przeniósł te koncepcje

[5]XVII-wieczny model tego systemu w postaci niewielkiej drewnianej skrzynki, zbudowanej na podstawie opisów znajdujących się w *Musurgia unversalis*, można dziś oglądać w bibliotece Magdalene College w Cambridge. Istnieje również jego współczesna, komputerowa realizacja, której dokonał amerykański programista Jim Bumgardner. Ciekawostkę może stanowić fakt, iż sposób przedstawiania danych w oryginalnym algorytmie zaproponowanym przez Kirchera, jest niemal identyczny ze sposobem ich reprezentacji w języku Perl, w którym program Bumgardnera został napisany [29].



Rysunek 2.2: *Arca Musarithmica*, XVII-wieczna maszyna do komponowania muzyki autorstwa Athanasiusa Kirchera [29, s. 2]

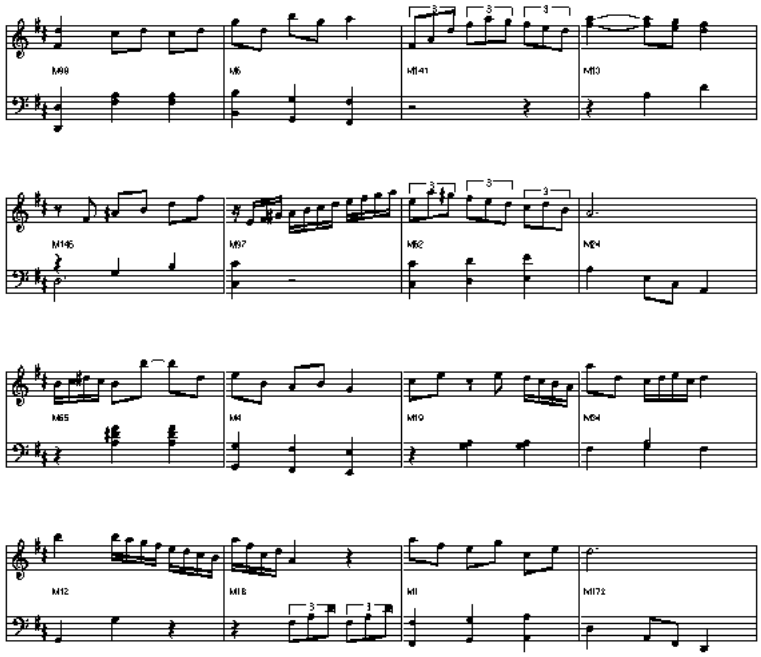
na muzykę, utrzymując że komponowanie muzyki polega na wynajdowaniu wciąż nowych kombinacji skończonej liczby elementów składowych podstawowego zbioru dźwięków. Nie trudno zauważyć ograniczenia, jakie implikują tego typu poglądy – zagadnienie to wróci jeszcze w rozważaniach dotyczących kompozycji komputerowej.

Koncepcję Kirchera kontynuowało wielu muzyków – w XVIII wieku powstało około 20 systemów automatycznego komponowania krótkich form muzycznych. Składały się one z elementów prekompozycyjnych, czyli gotowych fragmentów muzycznych, zestawu tablic oraz specjalnego bączka lub kości do gry [54, s. 68-70]. Zasady gry są tu bardzo proste: dla poszczególnych odcinków czasowych, którymi są najczęściej takty istnieje określona liczba układów muzycznych, z której można wybierać w sposób dowolny, bez ryzyka powstania błędów muzycznych. Liczba wymaganych dostępnych elementów odpowiadać musi liczbie oczek, np. przy dwóch kostkach powinno być 11 wariacji każdego taktu. Projektowanie tego typu systemu wymaga wzięcia pod uwagę nie tylko przebiegu harmonicznego, ale również sposobu prowadzenia głosów, tak aby we wszystkich możliwych kombinacjach powstawały układy

poprawne od strony technicznej. Pierwszym automatycznym systemem kompozycji tego typu było *Der allezeit fertige Menuetten- und Polonaisencomponist* Johanna Philippa Kirnbergera z 1757 roku. Wśród innych stworzonych przez wybitnych muzyków znajdują się m.in. *Einfall einen doppelten Contrapunct in der Octave von Sechs Takten zu machen ohne die Regeln davon zu wissen* Carla Philippa Emanuela Bacha z 1758 roku, czy *Table pour composer des minuets et des Trios à l'infinite: avec deux dez à jouer* Maximiliana Stadlera z około 1780 roku. Najsłynniejszy przykład wśród tego typu gier znajduje się bez wątpienia w opublikowej w 1792 roku książce pt. *Musikalisches Würfespiel*, której autorstwo przypisuje się Wolfgangowi Amadeuszowi Mozartowi. Przedstawiony tam system zawierał dwie tablice z przypisanymi do nich 176 taktami menueta i 96 taktami tria, których kolejność ustalało 16 rzutów dwiema kostkami do gry. Całkowita liczba wszystkich możliwych kombinacji dawała: $11^{16} * 6^{16} = 12962923816305025862428793241 6$ różnych menuetów[6] (rysunek 2.3).

W XX wieku technika serialna stanowiła kolejne wcielenie idei porządkowania materiału muzycznego przy pomocy procedur algorytmicznych. W istocie integralny serializm jest sposobem na automatyzację procesu kompozycji muzycznej, przez użycie elementów prekompozycyjnych, tj. "serie" i "matryce", zastosowanych do różnych parametrów muzycznych: wysokości dźwięków, wartości rytmicznych, dynamiki, artykulacji, czy barwy dźwięku.

[6]Implementację mozartowskiej gry w kości można znaleźć na stronie: http://sunsite.univie.ac.at/Mozart/dice/, dostęp: 2009-09-17.



Rysunek 2.3: Fragment jednego z menuetów wygenerowanych przy użyciu przypisywanej Mozartowi "gry w kości".

2.3.2 Komputerowo wspomagana kompozycja algorytmiczna

Idea wykorzystania komputera do tworzenia dzieł sztuki pochodzi od córki romantycznego poety George'a Byrona – Ady Augusty hrabiny Lovelace (1815-1852)[7] (rysunek 2.4). Uchodzi ona za autorkę pierwszego programu komputerowego, który stworzyła do sterowania teoretyczną *maszyną analityczną* Charles'a Babbagge'a [56, s. 104]. W swoich pochodzących z roku 1843 opisach tej wówczas nieosiągalnej z przyczyn technicznych maszyny, przedstawiła koncepcje fundamentalnych elementów współczesnych języków programowania: *pętli, podprogramu, instrukcji warunkowej.* Jednocześnie przewidziała, że maszyny będą w przyszłości zdolne do manipulowania nie tylko

[7]Jej imię nosi jeden z języków programowania stworzony w latach 70-tych dla potrzeb Departamentu Obronu USA przez Jeana Ichbiaha i zespół z CII Honeywell Bull [3].

liczbami, ale także symbolami, a w związku z tym można je będzie wykorzystać m.in. także do komponowania muzyki [69, s. xvi].

Rysunek 2.4: Ada Lovelace–autorka koncepcji muzyki komputerowej [4].

Gérard Assayag, jeden z głównych twórców programu *OpenMusic* pod pojęciem komputerowej muzyki algorytmicznej rozumie muzykę, będącą rezultatem wyliczeń komputerowych, opartych na danych wprowadzonych przez kompozytora.

> "Wstępna formalizacja umożliwia zaprogramowanie automatu, którego wynik działania jest uznawany bez żadnych ulepszeń, jako rezultat muzyczny. Podstawowym założeniem jest podział aktu kompozycji na aspekt techniczny i kreacyjny. Algorytm umożliwia zredukowanie trudności technicznych i ułatwienie kreacji. [21, s.2] "

W systemach tego typu nie ma możliwości kontrolowania przebiegu realizacji programu, a wynik jego działania uznaje się za gotowy utwór muzyczny. Od stopnia złożoności programu oraz zastosowanych w nim procedur kontrolnych zależy jak bardzo uzyskany rezultat odpowiada pierwotnej idei kompozytorskiej.

Pojawienie się komputerów w drugiej połowie XX wieku wyznaczyło nową erę w dziedzinie automatycznej kompozycji. Zaczęto wówczas stosować nowe paradygmaty teoretyczne, najczęściej przejęte z innych dziedzin wiedzy, tj. analiza statystyczna, lingwistyka, neurobiologia, itd. Należą do nich: modele Markowa, gramatyka formalna i generatywna, transition network, sztuczna inteligencja, fraktale i samopodobieństwo, algorytmy genetyczne, sztuczne sieci neuronowe, automaty komórkowe. Poniżej przedstawione zostały najważniejsze osiągnięcia w dziedzinie systemów komputerowo wspomaganej kompozycji algorytmicznej (CAAC), w których zastosowano niektóre z wyżej wymienionych paradygmatów.

Najwcześniejszy przykład zastosowania techniki komputerowej w kompozycji stanowi maszyna o nazwie "Electronic Music Synthesizer" Harry'ego F. Olsona i Henry'ego Belara. Została zbudowana na początku lat pięćdziesiątych, a opisana w artykule z 1961 roku. Składała się ona z dwu układów: jednego do generowania dźwięków oraz drugiego do kompozycji stochastycznej. Olson zanalizował jedenaście melodii Stephena Fostera, na bazie których stworzył łańcuchy Markowa pierwszego i drugiego stopnia w odniesieniu do wysokości dźwięków i rytmu [79, s. 71].

Pierwszą oryginalną kompozycją stworzoną przy użyciu komputera jest *Iliac Suite for String Quartet*, Lejarena Hillera oraz Leonarda Isaacsona z 1957 roku. Jej nazwa pochodzi od wykorzystanego w procesie kompozycji komputera ILLIAC I. Była to pierwsza maszyna licząca będąca własnością instytucji akademickiej – Uniwersytetu Illinois. Do wytwarzania materiału muzycznego Hiller zastosował algorytm Monte-Carlo[8] generujący liczby używane do kodyfikacji różnych parametrów muzycznych, tj. wysokości dźwięków, dynamika, ugrupowania rytmiczne, techniki instrumentalne. Parametry te w kolejnych częściach suity poddano regułom inspirowanym różnymi technikami kompozytorskimi: klasycznym kontrapunktem (cz.1), kontrapunktem swobodnym z kadencjami V-I (cz.2), techniką dodekafoniczną (cz.3) i aleatoryczną (cz.4). Implementacji tych reguł dokonano przy użyciu łańcuchów Markowa [60, s. 60].

[8]Twórcą metody Monte-Carlo był polski matematyk, współtwórca bomby termojądrowej Stanisław Ulam (1909-1984) [7].

Prace Pierre'a Barbauda i Rogera Blancharda z początku lat sześćdziesiątych, podobnie do eksperymentów Hillera, oparte były na metodach stochastycznych. Barbaud, autor pierwszego we Francji utworu muzyki komputerowej – *Factorielle 7*, zrealizowanego na komputerze *Gamma 3* firmy Honeywell Bull [9], poszedł dalej w formalizacji procesów muzycznych, używając do tego celu teorii zbiorów. Teoria ta była podstawą analizy języka tonalnego – zbiór wysokości był w niej przykładowo definiowany jako Z/12, zbiór pozostały jako modulo 12, a transpozycja jako dodawanie w Z/12. Według Barbauda kompozycja automatyczna jest rezultatem zastosowania różnych reguł na danych wyrażonych w formie zbiorów. Reguły te konkretyzowane były przy użyciu automatów o skończonej liczbie stanów lub matryc stochastycznych [21, s. 1].

Zastosowanie teorii zbiorów w kompozycji muzycznej odnajdujemy również w wielu kompozycjach Iannisa Xenakisa. W zaproponowanym przez niego ujęciu serie arytmetyczne mogły przykładowo służyć generowaniu nowych ukształtowań, dzięki zastosowaniu operacji na zbiorach. W myśl tej zasady operacja dodawania przeprowadzona na zbiorach A={0, 2, 4, 6, 8} i B={0, 3, 6, 9} daje w rezultacie nowy zbiór C={0, 2, 3, 4, 6, 8, 9}. Tego typu szeregi arytmetyczne mogą być używane w odniesieniu do różnych parametrów muzycznych: wysokości dźwięków, rytmów. Seria przedstawiona w powyższym przykładzie zastosowana do wysokości dźwięków daje, w przypadku wyboru skali chromatycznej oraz dźwięku wyjściowego C, następującą skalę: C, D, D#, E, F#, G#, A [79, s.45]. Xenakis wykorzystywał w celu realizacji tego typu procedur m.in. stworzony przez siebie w 1963 roku program *ST*. Napisany w języku Fortran, uruchomiony został w paryskim oddziale firmy IBM na komputerze IBM 7090. W przypadku Xenakisa, który w 1967 roku został profesorem "muzyki matematycznej i automatycznej" na Uniwersytecie Indiana, można powiedzieć, że używany przez niego język muzyczny w naturalny sposób korespondował z językiem maszyn. Kompozytor starał się bowiem manipulować zdarzeniami dźwiękowymi w oparciu o reguły zaczerpnięte z praw matematyki, logiki i fizyki [54, s. 83].

Ciekawą koncepcję podziału procesu kompozycji pomiędzy człowieka i komputer wprowadziły programy *PROJECT1* (1964) i *PROJECT2* (1966),

stworzone przez Gottfrieda Michaela Koeniga w Instituut voor Sonologie Uniwersytetu w Utrechcie. Program obliczał tu strukturę muzyczną, wychodząc od ustalonej specyfikacji jej ogólnego kształtu, dostarczonej przez kompozytora w postaci list parametrów opisujących: wartości rytmiczne, tempa, akordy itd. [2].

Jednym z najbardziej interesujących i znanych programów do automatycznego generowania muzyki jest bez wątpienia system *Experiments in Musical Intelligence* (EMI) Davida Cope'a[9]. Co więcej efekty uzyskane przez niego przy użyciu EMI, a także wyrażane przez niego poglądy prowokują nie tylko do rozważań na temat możliwości sztucznej inteligencji, ale również zmuszają do refleksji dotyczących istoty działalności twórczej. W tym rozbudowanym programie do generowania muzyki Cope zastosował reguły gramatyki formalnej zrealizowane przy użyciu techniki zwanej *augmented transition networks*. Dane wejściowe stanowi tu kilka przykładów utworów w określonym stylu muzycznym, które służą jako model do naśladowania. System skanuje dane w poszukiwaniu krótkich fragmentów charakteryzujących określony styl, a następnie automatycznie abstrahuje je i przekształca, tworząc szereg ich wariacji. W kolejnym etapie program tworzy kompletną kompozycję, budując ją kawałek po kawałku z przygotowanych wcześniej elementów [69, s. 74-75]. Głównym założeniem teoretycznym jest tu idea "rekombinacji" istniejącego materiału, stanowiąca sedno wyrażanych przez Cope'a poglądów:

> "Program ten stanowi paralelę do tego, co według mojej wiary dzieje się w mózgu kompozytora, świadomie lub nie. Wierzę, że geniusz wielkich kompozytorów leży nie w tworzeniu niewyobrażalnej wcześniej muzyki, lecz w ich zdolności efektywnego przekształcania i udoskonalania tego, co już istnieje [79, s. 123]."

Twierdzenie to stanowi echo opisanych wcześniej w tym rozdziale poglądów wyrażanych przez Athanasiusa Kirchera i wyraża jednocześnie istotę działa-

[9]Esej poświęcony temu zagadnieniu zatytułowany *Staring Emmy Straight in the Eye - and Doing My Best Not to Flinch* napisał Douglas Hofstadter, autor uhonorowanej nagrodą Pulitzera książki *Gödel, Escher, Bach: an Eternal Golden Braid*.

nia automatycznych systemów do komponowania muzyki, tj. *Arca Musarithmica* i XVIII-wieczne "muzyczne gry w kości". Imitacje muzyki Bacha, Mozarta, Beethovena, Chopina, Mahlera, czy Rachmaninowa stworzone przez EMI, prowokują do pytania, czy rzeczywiście wielka, oryginalna twórczość tych kompozytorów powstała jedynie w wyniku "rekombinacji"? Z drugiej strony rezultaty pracy Cope'a, rozpatrywane najczęściej w kontekście sztucznej inteligencji stanowią swoisty "muzyczny test Turinga"[10], którego rezultaty w przypadku EMI bywają niekiedy bardzo zaskakujące.

2.3.3 Systemy konwersacyjne

Systemy konwersacyjne to programy oferujące zestawy narzędzi przeznaczonych do rozwiązywania określonych problemów związanych z kompozycją muzyczną. Najważniejszą ich cechą jest możliwość wpływania na przebieg realizacji programu przed zakończeniem działania całego procesu obliczeniowego [60, s. 232]. Tego typu podejście odmienia sposób pracy z komputerem, który w początkowych latach rozwoju CAC był ukierunkowany przede wszystkim na zagadnienie formalizacji struktury muzycznej w oparciu o reguły matematyczne. We współczesnych systemach konwersacyjnych kompozytor uzyskuje wolność eksperymentowania w oparciu o intuicję, nie zaś z góry przyjęty plan formalny [13, s. xii].

Pierwszym komputerowym asystentem kompozytora był program *MUSICOMP* (MUsic Simulator-Interpreter for COMpositional Procedures), zaprojektowany przez Roberta Bakera i Lejarena Hillera około 1963 roku. Baker, będący wówczas studentem kompozycji, współpracował już wcześniej z Hillerem nad stworzeniem pierwszego komputerowego edytora nutowego – programu *Musicwriter*. *MUSICOMP* zawiera szeroki wybór procedur kompozytorskich w postaci zestawu ponad 40 podprogramów (ang. sub-routines), które łączone ze sobą mogą generować maksymalnie 31-głosowe struktury

[10]W jednej z wersji testu Turinga biorą udział dwie osoby znajdujące się w osobnych pokojach i komunikujące się ze sobą jedynie na piśmie. Początkowo chodzi o odgadnięcie płci osoby, której celem jest wprowadzenie testowanego w błąd. Turing sformułował nową wersję tej zabawy, w której pytanie brzmi: "Kto jest człowiekiem, a kto maszyną?" [79, s. 226].

muzyczne [23, s. 4]. Podprogramy te napisane w języku Fortran, stworzone były z myślą o rozwiązywaniu określonych problemów typowych dla kompozycji muzycznej. Dzielą się one na trzy grupy: generatory (głównie na bazie łańcuchów Markowa), modyfikatory serialne i geometryczne oraz zestaw reguł inspirowanych harmonią tradycyjną. Przy użyciu systemu *MUSICOMP* eksperymentowano m.in. z problemem organizacji rytmicznej w partiach instrumentów perkusyjnych oraz generowaniem muzyki serialnej przy użyciu modelu *Structures pour deux pianos* Pierre'a Bouleza. Inny kierunek rozwoju dotyczył badania struktur wertykalnych i horyzontalnych w skali temperowanej. Przy użyciu tego systemu stworzone zostały takie utwory Hillera jak: *Computer Cantata*, czy *HPSCHD* [21, s. 2].

Nowy etap w rozwoju oprogramowania CAC rozpoczyna się w połowie lat osiemdziesiątych. Wpływ na to miało kilka czynników, wsród których do najważniejszych należą bez wątpienia: pojawienie się komputerów osobistych, rozwój interfejsów graficznych, rozwój języków programowania, czy w końcu opracowanie w 1983 roku standardu MIDI (Cyfrowy Interfejs Instrumentów Muzycznych), który umożliwił łatwą komunikację między komputerem, a różnego typu elektronicznym sprzętem muzycznym. Powstaje wiele systemów, wśród nich m.in. stworzony w 1991 roku na uniwersytecie Stanforda *Common Music*, będący obiektowym środowiskiem programistycznym, umożliwiającym zarówno syntezę dźwięku, jak i algorytmiczne generowanie struktur muzycznych wyższego rzędu [102].

W ciągu ostatnich trzydziestu lat najważniejszym centrum rozwoju oprogramowania CAC stał się paryski IRCAM (Institut de Recherche et de Coordination Acoustique/Musique). Dzięki stałym dotacjom i instytucjonalnemu zarządzaniu, kilka zespołów programistów i muzyków pracowało tam z sukcesami nad opracowaniem i wdrożeniem nowych rozwiązań w dziedzinie zinformatyzowanej kompozycji. Pierwszym środowiskiem CAC rozwijanym w IRCAM-ie w latach 1982-85 był *FORMES*. Został stworzony przez Xaviera Rodeta i Pierre'a Cointe w języku VLisp i przeznaczony był do kompozycji i syntezy dźwieku. Składa się z zestawu klas, które opisują zachowanie się w czasie połączonych w hierarchiczną, rekursywną sieć procesów. Procesy te dzielą się na nadrzędne i podrzędne. Celem tych pierwszych jest odpo-

wiednia dystrybucja danych pomiędzy procesami podrzędnymi, koordynacja ich działania, a w końcu zebranie wyników obliczeń i przesłanie ich dalej do procesu głównego (root-process). Rezultatem działania programu są sygnały sterujące syntezatorami programowymi [21, s. 3].

Pierwszą próbą stworzenia środowiska programistycznego ogólnego zastosowania był natomiast program *Crime*, stworzony w latach 1985-86 w języku Lisp przez Gérarda Assayaga we współpracy z kompozytorem Claudym Malherbe. Użytkownik, dzięki dostarczonym mu językom formalnym, może tu definiować struktury rytmiczne, harmoniczne i polifoniczne o dowolnym stopniu komplikacji, a program przedstawia wyniki swojego działania w formie tradycyjnej notacji muzycznej. Stanowiło to ważną innowację, ułatwiającą współpracę kompozytora z komputerem. *Crime* zawiera również, po raz pierwszy wykorzystane w środowisku CAC modele psychoakustyczne, tj. analiza Terhardta służąca wyznaczaniu wirtualnego tonu podstawowego. Stosowało ją w tamtym czasie wielu kompozytorów m.in.: Claudy Malherbe, Marco Stroppa, Magnus Lindberg i Kaija Saariaho.

Kolejną ważną innowacją, zmierzającą w stronę modelu używanego przez większość dzisiejszych środowisk CAC, było wprowadzenie metody wizualizacji elementów składniowych i procesów kompozycyjnych. Jednym z pierwszych programów CAC wykorzystujących paradygmat *programowania wizualnego*, była napisana w języku PROLOG II *Carla*, stworzona w latach 1989-90 przez Francisa Courtota[11]. Środowisko to składa się ze zbioru podstawowych typów i powiązanej z nimi heurystyki, graficznego interfejsu dla formalizacji stosunków między typami oraz graficznego interfejsu dla programowania logicznego. *Carla* zawiera dwie klasy typów: pierwotne, definiowane przy użyciu języka logicznego pierwszego stopnia oraz złożone, które mogą być budowane z istniejących typów. Kompozytor przy użyciu graficznego interfejsu tworzy wykres koncepcyjny (graf), którego wierzchołki odpowiadają poszczególnym typom, a krawędzie reprezentują relacje między nimi.

Jednym z najważniejszych osiągnięć na polu informatyki muzycznej w końcu XX wieku było bez wątpienia powstanie języka *PatchWork*. Stwo-

[11]Paradygmat ten zastosowany został w tym okresie również w programie *Max*, a następnie w jego spadkobiercach: *jMax-ie* i *Pd*.

rzony w IRCAM-ie przez Mikaela Laursona, przy udziale Johna Duthena oraz Camilo Ruedy, oparty jest podobnie jak *Carla* na koncepcji programowania wizualnego. *PatchWork*, będący narzędziem programistycznym ogólnego zastosowania, jest w rzeczywistości graficznym interfejsem użytkownika (GUI) dla języka Lisp, w którym każda funkcja tego języka może być zamieniona w odpowiadający jej ekwiwalent wizualny – *box*. Poszczególne funkcje w *PatchWorku* mogą przechowywać chwilowe wartości i posiadają edytory graficzne umożliwiające dostęp do nich. Dzięki temu program posiada rozbudowaną bibliotekę predefiniowanych obiektów muzycznych, tj. *note, chord, chord-seq, poly*, wraz z odpowiadającymi im edytorami, dzięki którym dane oraz rezultaty obliczeń mogą być przedstawiane w formie notacji muzycznej. Jądro programu zaprojektowane jest w sposób neutralny od strony stylu muzycznego, w tym sensie, że nie jest oparte na żadnych wstępnych założeniach dotyczących rodzaju materiału, czy muzyki, która może być przy jego pomocy generowana lub analizowana. Cecha ta z jednej strony, a szeroki zakres procedur obliczeniowych, jaki dawał dostęp do funkcji języka Lisp z drugiej, zdecydowały o sukcesie *PatchWorku*. Dowodem na to jest fakt, iż program ten stał się w latach dziewięćdziesiątych XX-ego wieku niezwykle popularny wsród wybitnych kompozytorów o różnej orientacji stylistycznej, tj.: Brian Ferneyhough, Gérard Grisey, Paavo Heininen, Magnus Lindberg, Claudy Mahlerbe, Tristan Murail, czy Kaija Saariaho [15, s. 1-7].

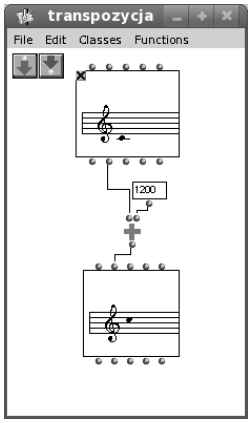
2.4 Narzędzia CAC wykorzystane w procesie komponowania utworu pt. *Rozbłyski*

Poniżej przedstawiona zostanie krótka charakterystyka programów komputerowych wykorzystanych podczas pracy nad utworem pt. *Rozbłyski*. Pierwszy z nich – *OpenMusic*, to rozbudowane środowisko programistyczne, będące jednocześnie językiem programowania oraz systemem konwersacyjnym służącym komputerowo wspomaganej kompozycji oraz analizie muzycznej. Drugi z nich – program *SPEAR*, to niewielkie, wyspecjalizowane narzędzie informatyczne, służące analizie spektralnej i resyntezie dźwięku. Obydwa stanowią

najnowocześniejsze obecnie narzędzia CAC.

2.4.1 OpenMusic

OpenMusic jest pod wieloma względami następcą *PatchWorku*, korzystającym z tej samej koncepcji programowania wizualnego oraz fragmentów jego kodu źródłowego. Jest on najnowszym systemem CAC stworzonym przez informatyków IRCAM-u: Carlosa Agona oraz Gérarda Assayaga. *OpenMusic* będąc obiektowym, graficznym językiem programowania zbudowanym w oparciu o Common Lisp oraz Common Lisp Object System, jest podobnie jak *PatchWork* środowiskiem programistycznym ogólnego zastosowania. Najważniejszą jego cechą jest *otwartość*, oznaczająca możliwość dowolnej konfiguracji, rozbudowy i swobodnej adaptacji do potrzeb użytkownika. Kompozytor lub teoretyk muzyki korzystający z *OpenMusic* ma możliwość rozbudowywania jego funkcjonalności, przez tworzenie nowych klas i funkcji w sposób czysto wizualny, bez konieczności zapisywania programu w postaci tekstowej. Podstawową jednostką programowania, a jednocześnie graficzną reprezentację programu w *OpenMusic* jest *łata* (ang. patch) (rysunek 2.5).



Rysunek 2.5: Okno programu *OpenMusic* przedstawiające prostą łatę realizującą transpozycję.

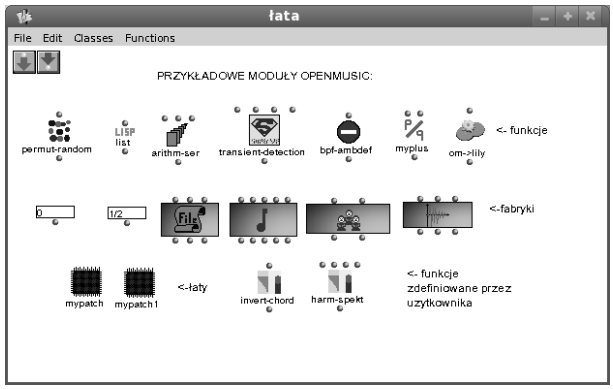
Projektowanie programu polega na umieszczaniu na powierzchni łaty ikon oraz ich odpowiednim łączeniu w celu stworzenia schematu blokowego repre-

zentującego kolejne czynności w programowanym algorytmie. Ikony w *OpenMusic* mogą odpowiadać trzem rodzajom obiektów (patrz rysunek 2.6):

1. **Funkcje**. Dzielą się na funkcję Common Lispa, wyspecjalizowane funkcje *OpenMusic* i funkcje zdefiniowane przez użytkownika. Dokonują różnego typu obliczeń, operując na przekazywanych do nich danych, przesyłając dalej wyniki. Wartościowanie funkcji nazywa się w nomenklaturze Lispa *ewaluacją*.

2. **Fabryki** (ang. factories) wytwarzają określonego typu obiekty, reprezentujące pewien rodzaj danych. Modelem struktury danych jest *klasa*. *OpenMusic* zawiera szeroki zestaw klas, które definiują różnego typu dane oraz związane z nimi metody. Są to m.in. nuta, akord, sekwencja akordów, plik midi oraz audio, liczba całkowita, ułamek, lista, itd.

3. **Łaty** mogą być używane w innych łatach jako oddzielne jednostki programowania realizujące określoną część obliczeń, pozwalając tym samym na budowanie bardzo złożonych programów. Mogą być one umieszczane także w sobie samych, tworząc tzw. *łaty rekurencyjne*[12].

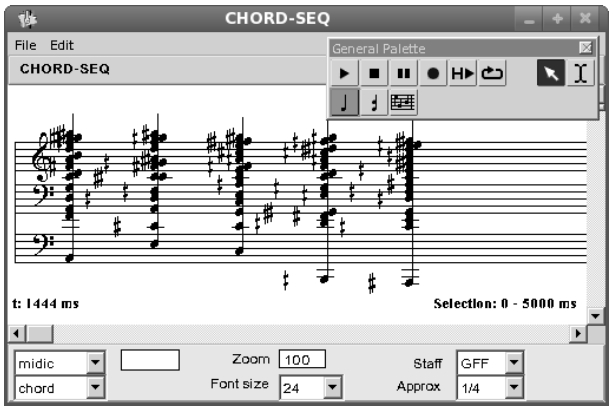


Rysunek 2.6: Okno programu prezentujące widok łaty z przykładowymi modułami *OpenMusic*.

Klasy muzyczne posiadają wyspecjalizowane edytory, umożliwiające dostęp do danych w celu ich inspekcji i manipulacji nimi. Notacja muzyczna

[12]Rekurencja lub rekursja to w logice, programowaniu i w matematyce odwoływanie się funkcji lub definicji do samej siebie [6].

w *OpenMusic* obsługiwana jest przez przenośną bibliotekę CMN (Common Music Notation) stworzoną przez Billa Schottstaedta w CCRMA. Ułatwia to pracę kompozytora, który może przedstawiać swoje pomysły w najbardziej naturalny dla siebie sposób, czyli za pomocą tradycyjnej notacji muzycznej (rysunek 2.7). Z drugiej strony poszczególne parametry dźwięku: wysokość, czas, dynamika, barwa (program GeneralMidi), przedstawiane są w postaci liczbowej, co umożliwia przeprowadzanie na ich wartościach różnych operacji matematycznych, jak przedstawione to zostało na rysunku 2.5.

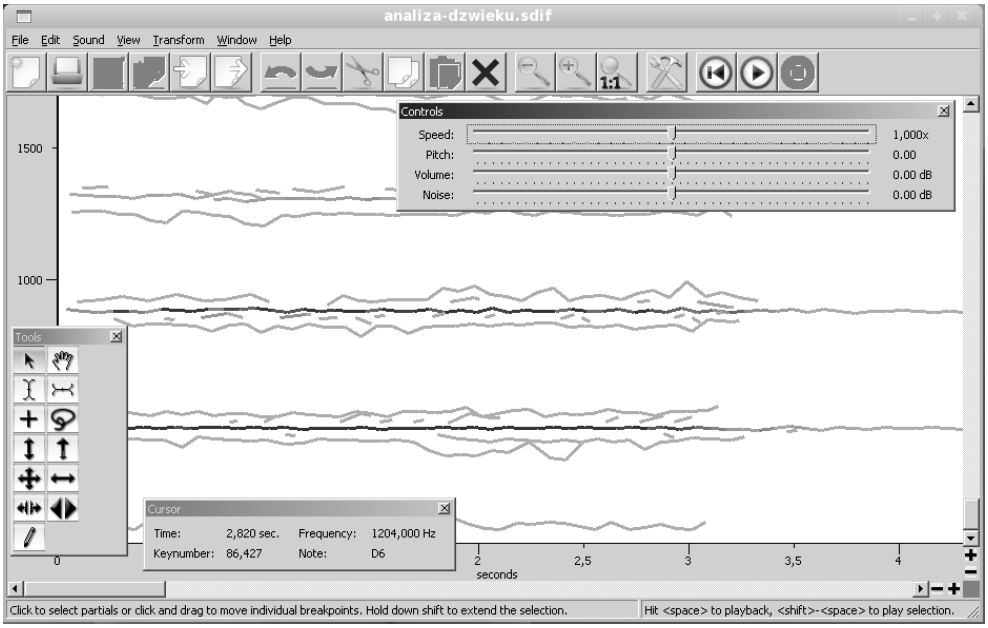


Rysunek 2.7: Edytor obiektu CHORD-SEQ prezentujący notację muzyczną w *OpenMusic*.

2.4.2 SPEAR

SPEAR to program przeznaczony do spektralnej analizy, edycji i syntezy dźwięku. Jego nazwa to akronim oznaczający Sinusoidal Partial Editing Analysis and Resynthesis. Do śledzenia składowych i interpolacji między szczytami amplitudy używa on techniki McAulay-Quatieri, dzięki której dokonywana jest możliwie najbardziej precyzyjna analiza dźwięku. Jednocześnie pozwala on na manipulowanie uzyskanymi rezultatami analizy umożliwiając ich zaznaczanie, kopiowanie, wklejanie, odwracanie itd., przy użyciu wyspecjalizowanego interfejsu graficznego, przypominającego programy do obróbki plików graficznych (rysunek 2.8). Przy jego użyciu można także w czasie rzeczywistym syntetyzować setki jednocześnie występujących składowych. *SPEAR* umożliwia importowanie i eksportowanie danych w różnych formatach, tj.:

różne odmiany formatu SDIF, ATS, czy pliki tekstowe [57].



Rysunek 2.8: Główne okno programu *SPEAR* przedstawiające rezultat analizy spektralnej dźwięku.

Rozdział 3

Analiza wybranych elementów języka dźwiękowego na przykładzie utworu pt. *Rozbłyski*

3.1 Zagadnienia ogólne

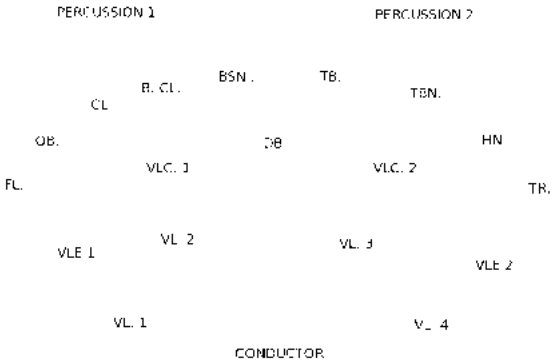
Język kompozytorski to niezwykle złożone zagadnienie obejmujące specyficzny dla danego kompozytora lub szkoły kompozytorskiej zbiór elementów oraz zespół reguł gramatycznych pozwalających łączyć je w większe całości. Pojęcie to związane jest w sposób nierozerwalny z pojęciem techniki kompozytorskiej, będącej zestawem procedur podejmowanych przez twórcę w celu skomponowania utworu muzycznego. Przebieg owych działań, ich kształt oraz logika ma swe źródło z jednej strony w świadomych działaniach kompozytora, wynikających z posługiwania się określonym aparatem czynności intelektualnych, z drugiej zaś w tym co możemy nazwać *fantazją* lub *inwencją*. Oczywiście ten drugi czynnik jest ze swej natury mało uchwytny i umyka analizie, stąd musi pozostać w ukryciu. Nie oznacza to jednak, że jest dla procesu powstawania dzieła mniej istotny. Można by zaryzykować stwierdzenie, że to właśnie działanie tej magicznej siły stanowi w powstawaniu każ-

dego utworu moment przełomowy, decydując w ostatecznym rozrachunku, nie tylko o kształcie dzieła, ale także o jego wartości.

Starając się opisać najważniejsze elementy swego języka dźwiękowego kompozytor musi z konieczności pozostać w sferze dostępnej analizie. Mając pełną świadomość cząstkowości tego typu opisu, a także trudności wynikających z faktu, że w tym przypadku badacz jest jednocześnie przedmiotem badania (co stawia pod znakiem zapytania zagadnienie obiektywizmu), autor przystępuje do przedstawienia analizy wybranych aspektów utworu pt. *Rozbłyski*. Poniżej dokonany zostanie opis obsady instrumentalnej oraz szczegółowo przedstawiona zostanie organizacja materiału dźwiękowego w aspekcie wysokościowym, jako ten element języka dźwiękowego autora, w którym komputerowo wspomagana kompozycja znalazła zastosowanie w największym stopniu. Na zakończenie forma utworu zostanie ukazana w kontekście organizacji wysokościowej, dopełniając w ten sposób całościowy obraz utworu.

3.2 Obsada instrumentalna

Aparat wykonawczy jest zawsze jednym z kluczowych czynników determinujących ostateczny kształt kompozycji. Jednocześnie inspiruje i sugeruje użycie określonych rozwiązań muzycznych. W utworze pt. *Rozbłyski* wykorzystano skład małej orkiestry symfonicznej: pojedynczą obsadę instrumentów dętych (fl., ob., cl., cl.b., fg., tr., hn., tbn., tb.) wzmacnia dwóch perkusistów obsługujących 13 instrumentów (glk., 3 cymb., snare drum, tom-toms, wood-blocks, vibraslap, bass-drum, vibr., 2 gongs, tam-tam, bar chimes) oraz poszerzony do dziewięciu głosów zestaw instrumentów smyczkowych (4 vl, 2 vle, 2 vc, 1 db). W celu uzyskania efektów przestrzennych i uplastycznienia zawartych w partyturze efektów akustycznych (dialogi instrumentów, efekty echa i pogłosu) zastosowano specjalny rozkład instrumentalistów na estradzie (rysunek 3.1).



Rysunek 3.1: Schemat rozmieszczenia instrumentalistów na estradzie podczas wykonania utworu pt. *Rozbłyski*.

3.3 Organizacja materiału muzycznego w aspekcie wysokościowym

Założenia teoretyczne, będące fundamentem organizacji materiału muzycznego w aspekcie wysokościowym w utworze *Rozbłyski*, oparte są na zdobyczach tzw. *francuskiej szkoły spektralnej*. Za jej prekursorów, a jednocześnie najwybitniejszych przedstawicieli uznaje się przede wszystkim Gérarda Griseya (1946-1998), Tristana Murailа (ur. 1947) oraz Huguesa Dufourta (ur. 1943), a do grupy tej zalicza się także przedstawicieli młodszego pokolenia, tj.: Kaija Saariaho (ur. 1952), Philippe Hurel (ur. 1955) czy Marc-André Dalbavie (ur. 1961). Choć każdy z tych twórców reprezentuje inny typ osobowości artystycznej, to można powiedzieć, że najważniejszym wyznacznikiem stylu ich muzyki jest, jak to ujął sam Grisey: "zaproponowanie organizacji formalnej i materiału dźwiękowego wywodzących się bezpośrednio z fizyki fal akustycznych badanych naukowo w skali mikro"[50, s. 2]. Muzyka spektralna wykształciła zupełnie nowy, niezwykle oryginalny język dźwiękowy, którego kluczowym elementem jest, zdaniem autora, specyficzny rodzaj kategorii brzmieniowej: współbrzmienia istniejące na styku dwóch wcześniej rozłącz-

nych pojęć: harmonii i barwy[1]. Równie istotne jest także zastąpienie opozycji konsonans-dysonans kategoriami harmoniczności-nieharmoniczności. Język dźwiękowy kompozytorów spektralnych cechuje często (choć nie jest to warunek niezbędny) odejście od dwunastodźwiękowej skali temperowanej. Mikrotonowość w muzyce spektralnej nie wynika jednakże z użycia określonej skali (np. ćwierćtonowej), lecz jest sposobem na przybliżenie określonej częstotliwości do najbliższej używanej w praktyce instrumentalnej [43, s. 84]. W utworze pt. *Rozbłyski* zastosowano materiał dźwiękowy oparty na naturalnych fenomenach akustycznych, użyte zostały także niektóre techniki zaproponowane przez spektralistów. W odróżnieniu od tych kompozytorów autor, biorąc pod uwagę skład instrumentalny zdecydował się jednakże na zastosowanie materiału dźwiękowego temperowanej skali dwunastodźwiękowej.

W dalszej części rozdziału przedstawiony zostanie szczegółowy opis organizacji materiału dźwiękowego, wykorzystanego w utworze pt. *Rozbłyski*, w aspekcie wysokościowym. Materiał ten można podzielić na kilka kategorii:

- struktury oparte na szeregu harmonicznym
- struktury oparte na szeregu subharmonicznym
- struktury powstałe w wyniku manipulowania szeregami harmonicznymi i subharmonicznymi
- akordy FM
- akordy "konkretne"

3.3.1 Struktury oparte na szeregu harmonicznym

Szereg harmoniczny jest matematyczną reprezentacją zjawisk dźwiękowych, określanych w akustyce mianem *wielotonów harmonicznych*[2]. Składa się z

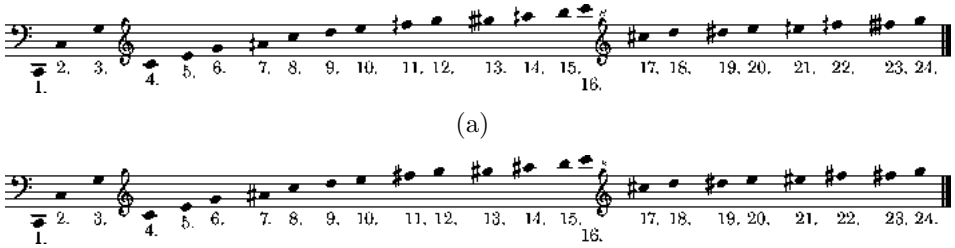
[1]Prekursorami tego typu podejścia do zagadnień harmonii i barwy byli m.in. Claude Debussy, Edgar Varèse i György Ligeti [35]

[2]Powstają one m.in. w instrumentach strunowych, dętych, czy głosie ludzkim. Czysty szereg harmoniczny jest w pewnym sensie konstrukcją teoretyczną, gdyż dźwięki instrumentów muzycznych nie są nigdy idealnie harmoniczne. Posiadają składniki szumowe wywołane np. oddechem grającego na instrumencie dętym, czy szuraniem smyczka po

ciągu wartości stanowiących wielokrotności najniższej częstotliwości, określanej mianem tonu podstawowego. Poniżej przedstawione są częstotliwości (wyrażone w Hz) kolejnych 24 tonów składowych szeregu harmonicznego dźwięku C:

1. 65.40639	7. 457.62637	13. 850.0239	19. 1243.071
2. 130.81277	8. 523.2511	14. 915.25275	20. 1307.8907
3. 196.22429	9. 588.6881	15. 980.9436	21. 1372.9148
4. 261.62555	10. 653.9454	16. 1046.5022	22. 1439.5077
5. 326.9727	11. 719.7538	17. 1111.2953	23. 1504.1086
6. 392.44855	12. 784.8971	18. 1177.3762	24. 1569.7943

Ciąg ten zawiera częstotliwości, które nie mieszczą się w ramach dwunastodźwiękowego systemu temperowanego, stąd zachodzi potrzeba przybliżenia ich do najbliższej używanej wysokości. Poniżej szereg ten przedstawiony został w postaci notacji muzycznej z wykorzystaniem skali ćwierćtonowej (a) oraz półtononowej (b):[3]:



(a)

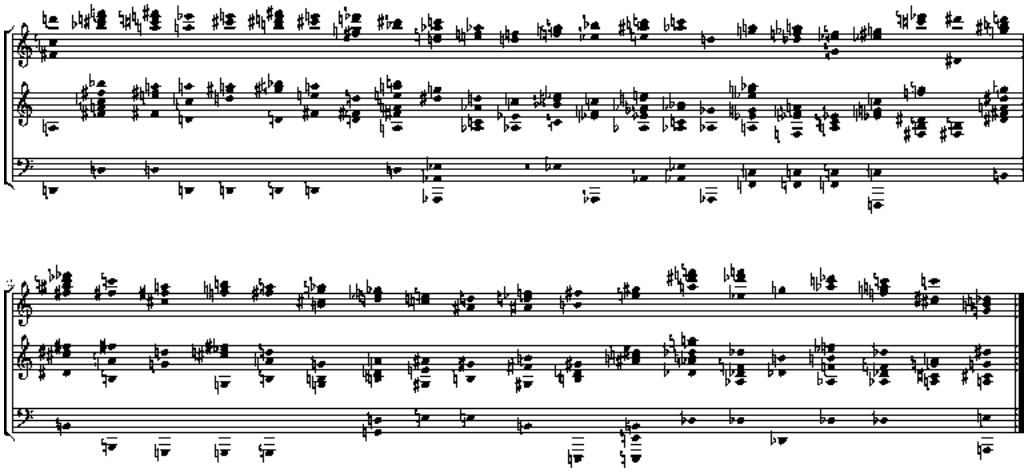
(b)

Rysunek 3.2: Pierwsze 24 alikwoty dźwięku C (a) w przybliżeniu do 1/4 tonu oraz (b) w przybliżeniu do 1/2 tonu.

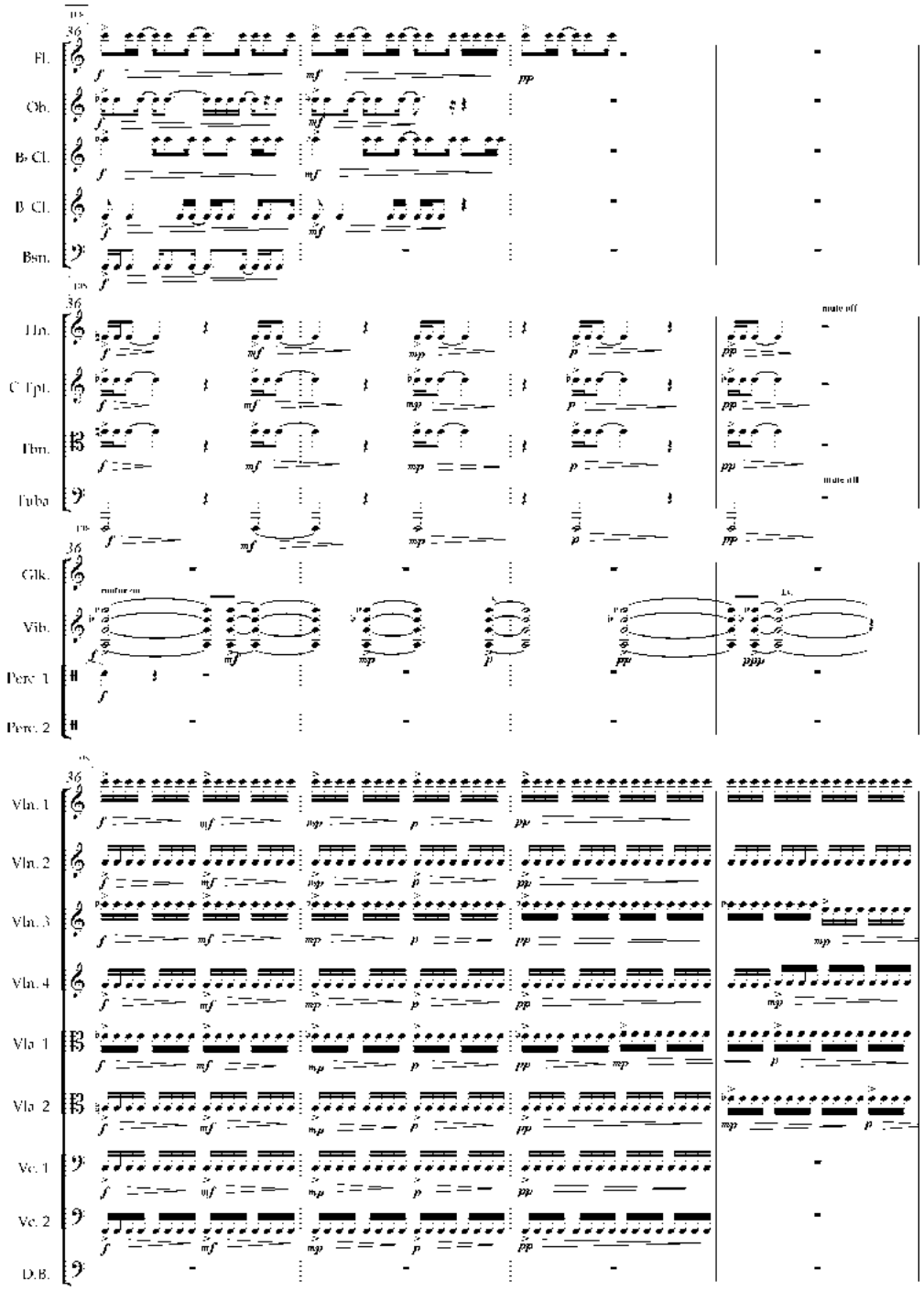
strunie. Oprócz tego układ kolejnych składowych bywa czasem nieznacznie ściśnięty lub rozciągnięty (tak jak w dźwięku fortepianu), szeregi te mogą być niepełne (np. dźwięk klarnetu zawiera tylko nieparzyste alikwoty), a ilość składowych jest bardzo różna i zmienia się w czasie [43, s. 85-87].

[3]Ten oraz kolejne przykłady nutowe zostały stworzone przy użyciu programu *OpenMusic*, biblioteki *omlily* oraz programu *LilyPond*.

Opierając się na opisanym powyżej modelu oraz wykorzystując narzedzie CAC przedstawione w podrozdziale 4.1.3 na stronie 69, stworzony został materiał dźwiękowy złożony w całości ze współbrzmień odzwierciedlających strukturę szeregu harmonicznego. Spektrum harmoniczne dźwięku D, obejmujące składowe od 1. do 20. zostało zmultiplikowane, następnie zaś wielkokrotnie i za każdym razem w inny sposób przefiltrowane, a w końcu przetransponowane na dźwięki $As_1, F_1, H_1, G_1, E_1, Des, A_1$. Stanowią one realne lub wirtualne tony podstawowe uzyskanych spektrów. W wyniku tych czynności uzyskano progresję akordów przedstawioną na rysunku 3.3. Ukazuje on materiał dźwiękowy stanowiący harmoniczną podstawę I części utworu pt. *Rozbłyski – Adagio assai*. Przykład wykorzystania tego materiału przedstawiono z kolei na rysunku 3.4. Przedstawia on fragment partytury (t. 36–39), w którym widoczny jest materiał dźwiękowy spektrum harmonicznego dźwięku F_1, czyli akordów nr 18-20 z rysunku 3.3.



Rysunek 3.3: Progresja harmoniczna złożona z ciągu filtrowanych i transponowanych spektrów harmonicznych, wygenerowana przy użyciu programu przedstawionego na rysunku 4.9 na stronie 70.



Rysunek 3.4: Fragment partytury I części –*Adagio assai* utworu pt. *Rozbłyski* oparty w całości na materiale spektrum harmonicznego dźwięku F_1.

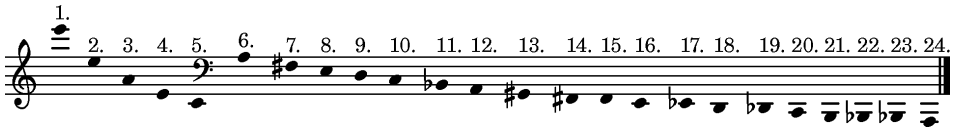
3.3.2 Struktury oparte na szeregu subharmonicznym

Szereg harmoniczny charakteryzuje się systematycznym zmniejszaniem się odległości pomiędzy kolejnymi, coraz wyższymi składnikami. Odwracając (w sposób arbitralny) porządek interwałów uzyskamy sztuczny twór – szereg subharmoniczny, zwany również szeregiem dolnym[4]. Poniżej przedstawione są wartości (wyrażone w Hz) kolejnych składowych ciągu subharmonicznego dźwięku e^3:

1. 1318.5103	7. 188.44849	13. 101.45479	19. 69.37576
2. 659.2551	8. 164.81378	14. 94.22424	20. 65.93747
3. 439.49197	9. 146.49352	15. 87.91434	21. 62.81453
4. 329.62756	10. 131.87492	16. 82.40688	22. 59.90867
5. 263.74985	11. 119.81735	17. 77.60224	23. 57.335613
6. 219.74599	12. 109.872986	18. 73.246765	24. 54.936493

A tak wygląda ten szereg przedstawiony w postaci zapisu nutowego (dźwięki zostały przybliżone do 1/2 tonu):

[4]Struktura szeregów: górnego i dolnego od dawna służyła teoretycznym rozważaniom dotyczącym różnych systemów dźwiękowych. Jean–Phillippe Rameau opierając się na badaniach głuchego od urodzenia matematyka i akustyka (!) francuskiego Josepha Sauveura (1653-1716), w szeregu górnym widział uzasadnienie opisywanego przez siebie systemu dur-mol. W wydanym w roku 1722 słynnym *Traité de l'Harmonie réduite à ses principes naturels* wyjaśniał pochodzenie akordu durowego i dominanty septymowej od składowych tego szeregu. Usankcjonowanie funkcji subdominanty oraz wyjaśnienie pochodzenia akordu molowego na podstawie szeregu dolnego pojawiły się u Rameau w traktacie *Génération harmonique* w roku 1737. Rameau uzyskał szereg dolny przez zestawienie strun, z których każda kolejna posiadała długość 2, 3, 4 itd. razy dłuższą od pierwszej. Te dłuższe struny nie wydawały wprawdzie dźwięku, lecz jak twierdził Rameau – drgały. "Na szeregu górnym, ujawniającym się w wibrujących i brzmiących strunach, opiera się dominanta, królestwo radości, światła i siły; na szeregu dolnym, ujawniającym się poprzez bezdźwięczne, a jedynie drżące struny – subdominanta, królestwo smutku, mroku i słabości" (J.-Ph. Rameau, *Observations sur notre instinct pour la musique*, 1754, cyt. za: [46, s. 521]).



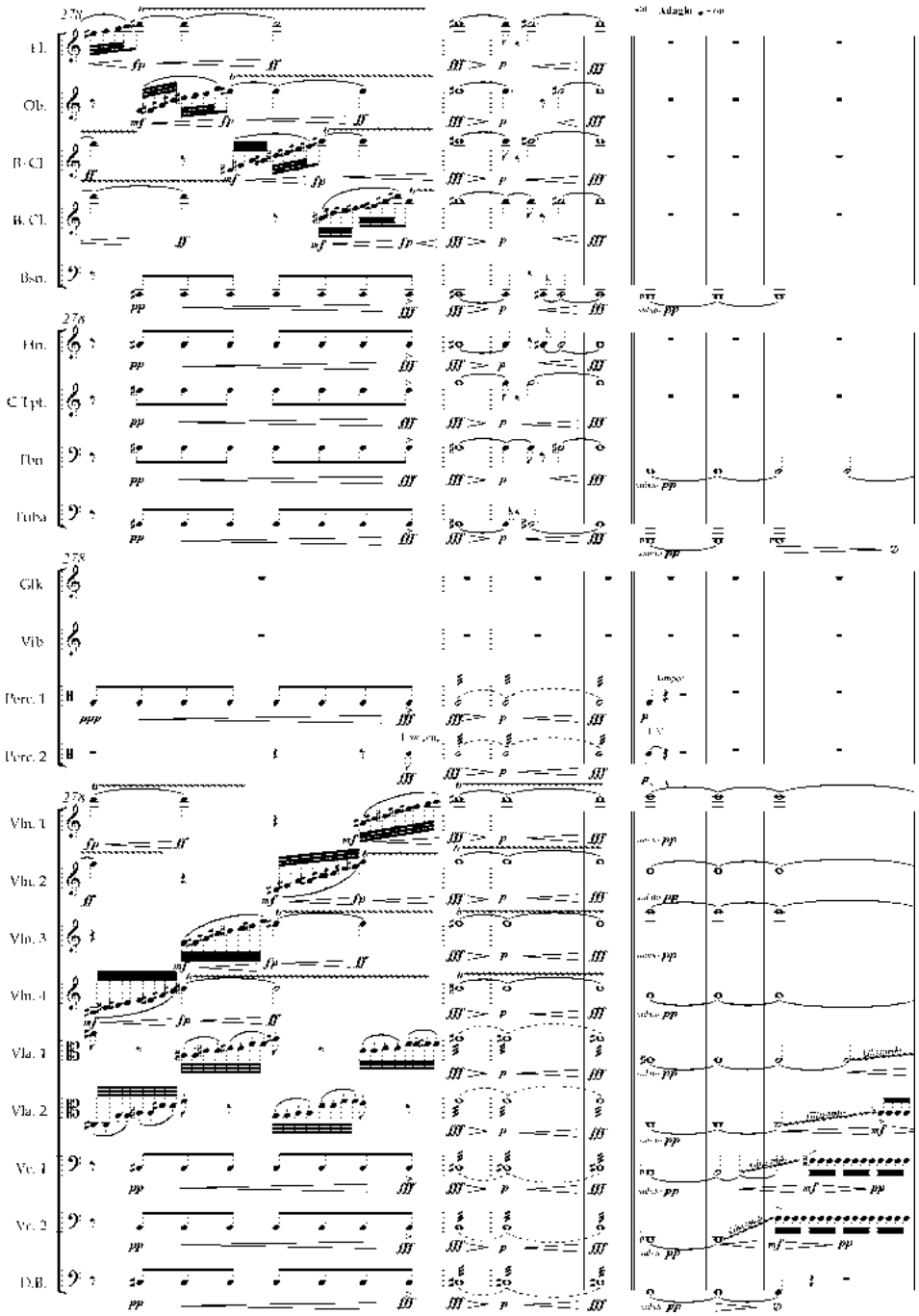
Rysunek 3.5: Odwrócony szereg harmoniczny (tzw. szereg dolny) dźwięku e^3.

Szereg subharmoniczny, a także materiał powstały w wyniku manipulowania nim stworzony został przy użyciu programu opisanego w podrozdziale 4.1.6 na stronie 74. Materiał ten wykorzystany został na początku ostatniej – czwartej części utworu pt. *Rozbłyski* (rysunek 3.6). Użycie tego właśnie materiału w tym miejscu kompozycji nie jest przypadkowe. Część ta stanowi bowiem antytezę części pierwszej, opartej na strukturach wynikających z szeregu harmonicznego. Użycie współbrzmienia odzwierciedlającego szereg subharmoniczny jest silnym akcentem ekspresyjno–brzmieniowym, wyznaczającym jeden z kluczowych punktów rozwoju formy.

3.3.3 Struktury powstałe w wyniku manipulowania szeregami harmonicznymi i subharmonicznymi. Rotacja akordów

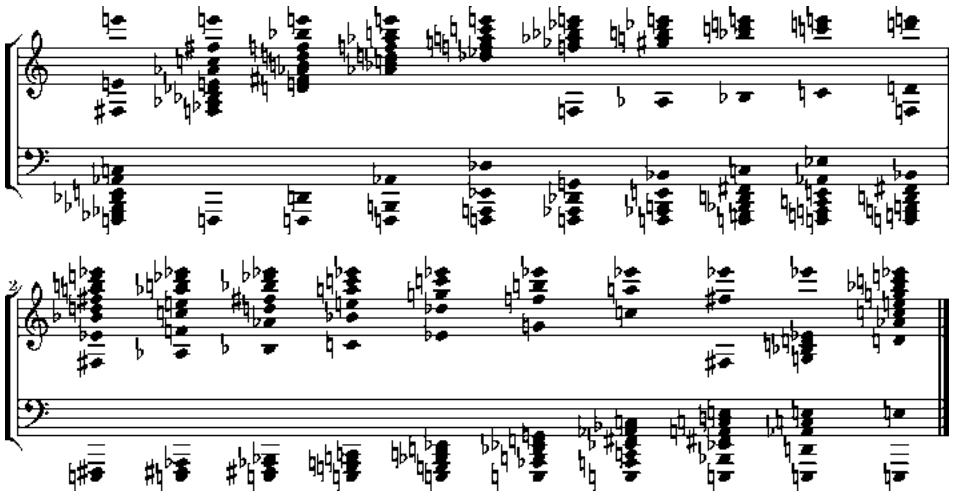
Szeregi: harmoniczny i subharmoniczny poddane zostały procedurze technicznej, którą na potrzeby tej pracy nazwiemy *rotacją akordu*. Przez rotację rozumie się w tym przypadku specjalny rodzaj permutacji, prowadzący do przekształcenia struktury wyjściowej i uzyskania szeregu struktur pochodnych. W rotacji dźwięki skrajne akordu zostają zachowane jako dźwięki stałe, podczas gdy kolejność interwałów jest stopniowo przesuwana. Jeśli odległości między kolejnymi dźwiękami w danym współbrzmieniu przedstawimy jako: (a, b, c), to kolejne rotacje będą miały postać: (b, c, a) oraz (c, a, b).

Technika rotacji umożliwia uzyskanie bogatego materiału dźwiękowego, charakteryzującego się spójnością, którą zapewnia wykorzystanie fragmentów tej samej struktury wyjściowej, ukazywanej za każdym razem w innej permutacji. Rysunek 3.7 przedstawia progresję harmoniczną, uzyskaną w wyniku działania programu opisanego w podrozdziale 4.1.6 na stronie 74. W górnym systemie widzimy szereg subharmoniczny dźwięku e^3, następnie zaś dziewieć

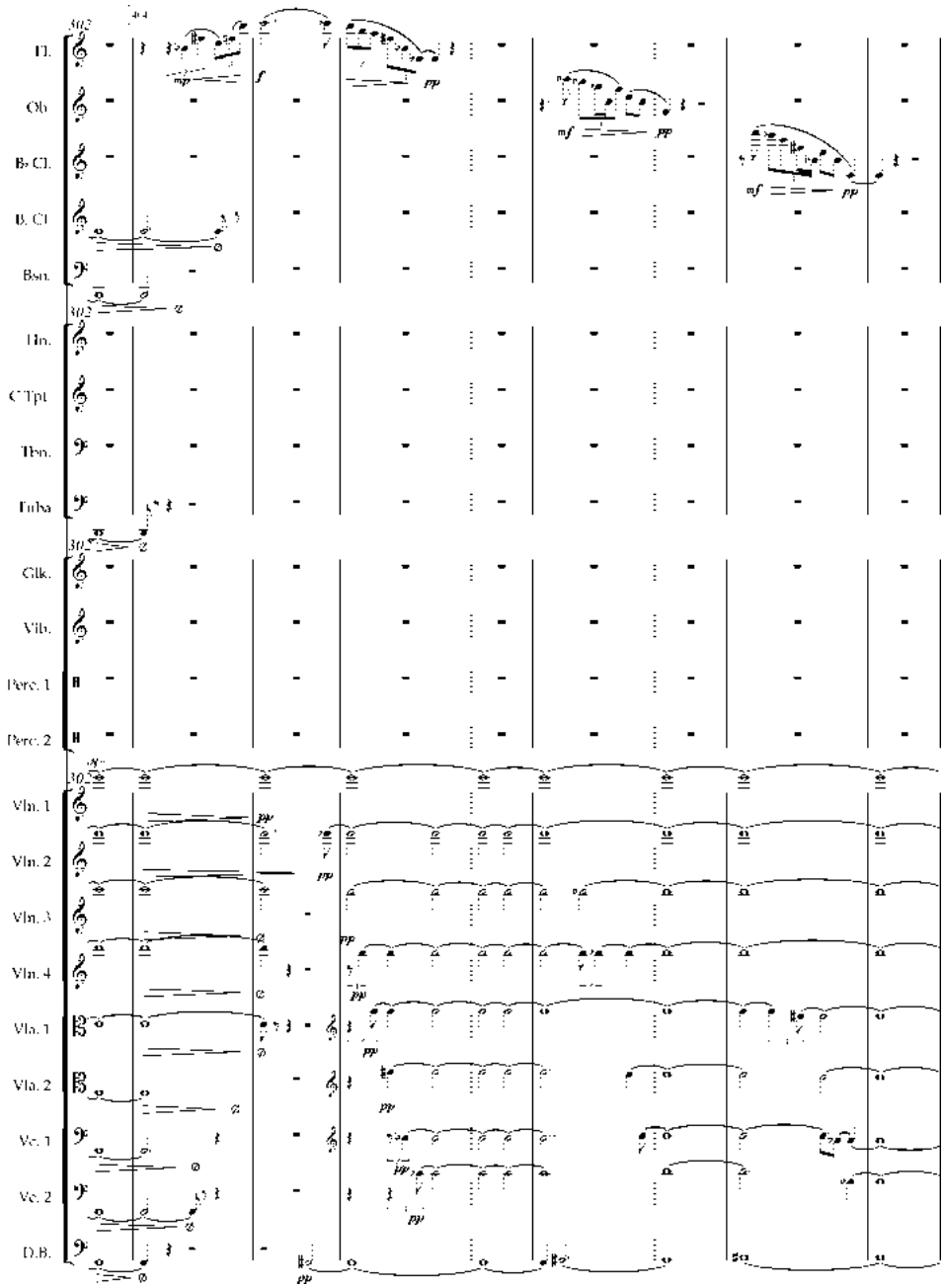


Rysunek 3.6: Fragment partytury utworu pt. *Rozbłyski*, ukazujący początek ostatniej części – *Adagio*. Współbrzmienie oparte na szeregu subharmonicznym dźwięku e^3 widoczne w t. 282-284

jego kolejnych rotacji. W dolnym systemie obserwujemy odwróconą kolejność przemian, gdyż najpierw następują rotacje, na końcu zaś pojawia się struktura wyjściowa, czyli spektrum harmoniczne dźwięku E_1. Materiał ten został użyty w ostatniej części utworu pt. *Rozbłyski - Adagio*, co widzimy na rysunku 3.8. Zarówno szereg subharmoniczny, jego rotacje oraz rotacje szeregu harmonicznego charakteryzują się nieharmonicznością. Ostatni widoczny na rysunku 3.7 akord, będący spektrum harmonicznym dźwięku E_1, nie pojawia się w partyturze w sposób dosłowny. Spektrum to jest jednakże słyszalne w niskim dźwięku, granym przez kontrabasy w taktach 317-322. Zastosowanie materiału o charakterze nieharmonicznym w ostatniej części utworu pt. *Rozbłyski* było zabiegiem formalnym oraz ekspresyjnym i jak już była o tym mowa wcześniej, stanowi antytezę części pierwszej, opartej na spektrach harmonicznych.



Rysunek 3.7: Progresja harmoniczna uzyskana w wyniku rotacji szeregu subharmonicznego dźwięku e^3 i harmonicznego dźwięku E_1.



Rysunek 3.8: Fragment partytury (t. 302–310) ostatniej części -*Adagio* utworu pt. *Rozbłyski*, ukazujący struktury harmoniczne powstałe w wyniku rotacji spektrum harmonicznego.

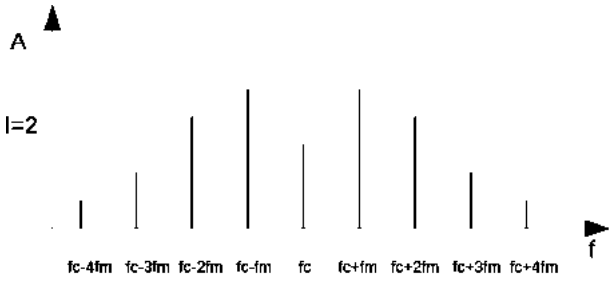
3.3.4 Akordy FM

Akordy FM (ang. *FM chords*) to struktury harmoniczne oparte na modelu syntezy przez modulację częstotliwości (ang. *Frequency Modulation*). Modulacja częstotliwości należy do nieliniowych przekształceń sygnału i powstaje w wyniku oddziaływania na sygnał nośny (ang. *carrier*) innym sygnałem zwanym modulatorem (ang. *modulator*). Modulację częstotliwości stosuje się m.in. w radiofonii oraz syntezie dźwięku (zarówno analogowej, jak i cyfrowej). Jedno z najciekawszych zastosowań modulacji częstotliwości stanowi metoda cyfrowej syntezy złożonych widm dźwiękowych stworzona przez Johna M. Chowninga, kierownika Center for Computer Research in Music and Acoustics na Stanford University. Służy ona do imitowania dźwięków instrumentów naturalnych oraz tworzenia dźwięków syntetycznych o dużej złożoności. Ponieważ zastosowanie modulacji częstotliwości w utworze pt. *Rozbłyski* nie dotyczyło komputerowej syntezy dźwięku, pominięte tu zostaną wyjaśnienia dotyczące szczegółów metody zaproponowanej przez Chowninga[5]. Inne zastosowanie modulacja częstotliwości znalazła także w kompozycji instrumentalnej, stając się modelem dla tworzenia tzw. *akordów FM*[6], czyli współbrzmień których struktura powstaje w wyniku symulacji zjawisk fizycznych zachodzących w modulacji częstotliwości. Rezultatem jej jest bowiem złożone spektrum, składające się z częstotliwości nośnej f_c oraz szeregu częstotliwości różniących się od niej o kolejne wielokrotności częstotliwości modulującej f_m w górę i w dół. Składowe te, zwane *wstęgami bocznymi*, układają się w następujący szereg (jego ilustrację stanowi rysunek 3.9):

$$f_c - nf_m, ..., f_c - 2f_m, f_c - f_m, f_c, f_c + f_m, f_c + 2f_m, ..., f_c + nf_m$$

[5]Mówiąc najkrócej, metoda ta opiera się na pewnej właściwości modulacji częstotliwości decydującej o tym, że w przypadku gdy sygnał nośny i modulujący znajdują się w paśmie słyszalności, szerokość widma zmodulowanego sygnału zmienia się w zależności od tzw. wskaźnika modulacji (ang. *modulation index*), przy czym układ częstotliwości pozostaje niezmieniony. Nadaje to syntezowanym dźwiękom charakter zbliżony do dźwięków naturalnych. Dokładniejsze wyjaśnienie tej metody znajduje się m.in. w [60, s. 241-249], [82, 63-68] oraz [34].

[6]*Akordy FM* zastosowane zostały po raz pierwszy przez Tristana Murailа w utworze orkiestrowym pt. *Gondwana* i stały się jednym z podstawowych środków harmonicznych w muzyce spektralnej [86, 43].

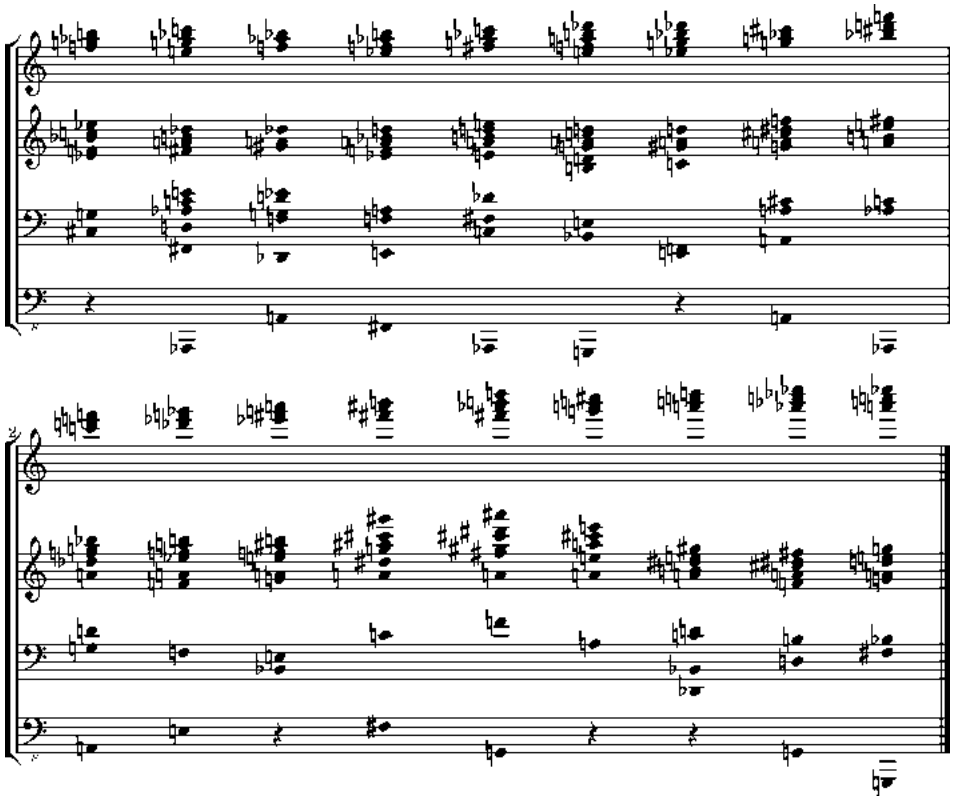


Rysunek 3.9: Wykres przedstawiający wstęgi boczne ułożone symetrycznie wokół częstotliwości nośnej [60, s. 243].

Model syntezy FM może dostarczyć wielkiego bogactwa struktur harmonicznych o różnej charakterystyce brzmieniowej. Jak podaje Włodzimierz Kotoński [60, s. 247], w zależności od stosunku częstotliwości nośnej do częstotliwości modulującej, rezultatem modulacji częstotliwości mogą być wielotony o strukturze odzworowującej widma: harmoniczne (prosty stosunek f_c/f_m), nieharmoniczne (bardziej złożone relacjef_c/f_m, np. wyrażone liczbą niewymierną), harmoniczne, ale "brzmiące, jak nieharmoniczne" (gdy podstawa widma harmonicznego leży poniżej pasma słyszalności, gdy w widmie brak pierwszych składowych lub znajduje się w nim duża liczba składowych "odbitych"[7]).

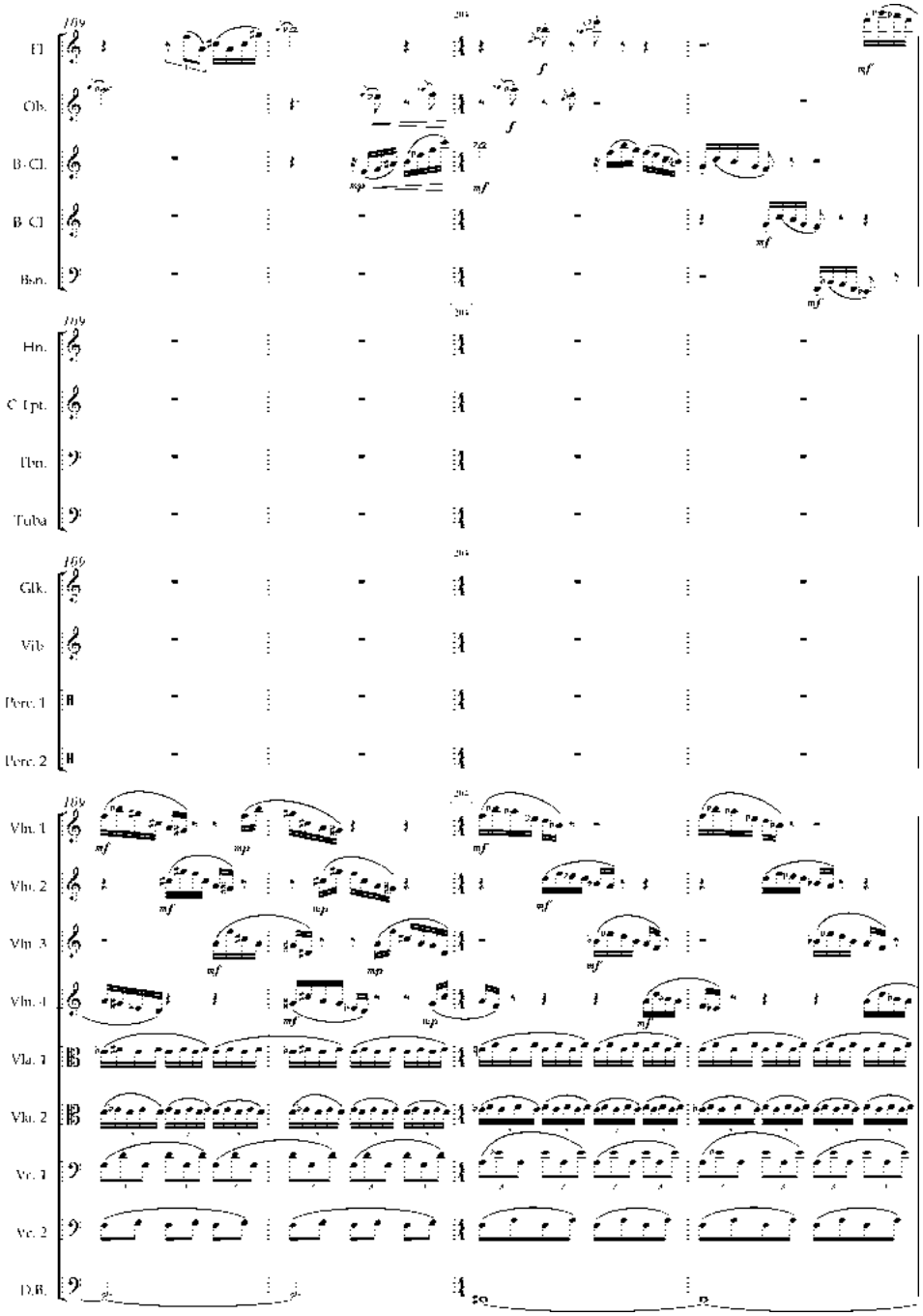
Akordy FM stworzone zostały przy użyciu programu opisanego w podrozdziale 4.1.4 na stronie 71. Podstawą dla wyznaczenia progresji harmonicznej akordów FM, wykorzystanej w utworze pt. *Rozbłyski*, była dwugłosowa struktura, w której górny głos będący częstotliwością nośną stanowiła nuta stała a^1, zaś głos dolny reprezentujący kolejne częstotliwości modulujące stanowił pochód chromatyczny od B do gis^1. Uzyskany w ten sposób szereg akordów został następnie posegregowany (niektóre z akordów zostały odrzucone), częściowo na drodze intuicyjnej, częściowo przy użyciu narzędzi informatycznych, o których mowa w podrozdziale 4.3.1 na stronie 82. W wyniku tych czynności uzyskano progresję harmoniczną przedstawioną na rysunku 3.10.

[7]Składowa odbita powstaje wtedy gdy wynik uzyskany z odejmowania $f_c - (n)f_m$ przyjmuje wartość ujemną. Czestotliwość taka przybiera wówczas wartość absolutną, lecz sygnał taki ma fazę odwróconą o 180°.



Rysunek 3.10: Progresja akordów FM stanowiąca podstawę drugiej części - *Allegro energico. Moderato* utworu pt. *Rozbłyski*.

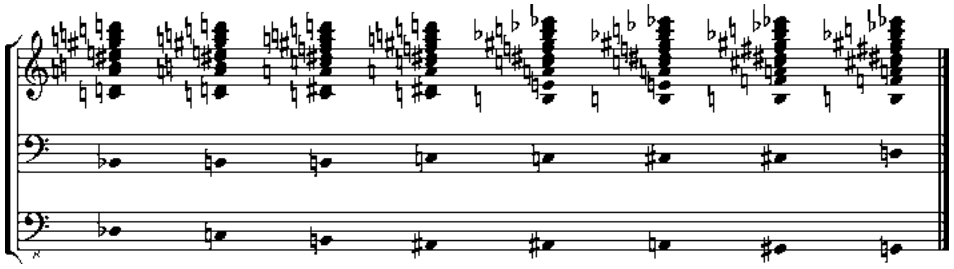
Materiał dźwiękowy akordów FM użyty został w drugiej części utworu pt. *Rozbłyski*, czyli *Allegro energico. Moderato*. Na rysunku 3.11 widzimy fragment partytury, w którym wykorzystano akordy FM



Rysunek 3.11: Fragment partytury (t. 109–112) drugiej części –*Allegro energico* utworu pt. *Rozbłyski*, przedstawiający sposób wykorzystania akordów FM.

3.3.5 Interpolacje harmoniczne

Pod pojęciem interpolacji[8] harmonicznych rozumie się taki rodzaj stopniowej transformacji materiału, który prowadzi do wytworzenia nowych układów dźwiękowych przez wyznaczenie stopni pośrednich, odpowiadających matematycznemu pojęciu *węzłów funkcji*, pomiędzy dwoma strukturami: wyjściową i docelową[9]. Rysunek 3.12 przedstawia progresję akordów uzyskaną przy użyciu techniki interpolacji i wykorzystaną w części II – *Moderato*, w taktach 205-243 (patrz rysunek 3.13). Struktury: wyjściową i docelową stanowią tu dwa akordy FM – widoczne na rysunku 3.10 jako trzeci i drugi od końca. Progresja ta stworzona została przy użyciu programu opisanego w podrozdziale 4.1.5 na stronie 73. W tym samym fragmencie kompozycji w partiach instrumentów dętych drewnianych oraz wibrafonu zastosowano również materiał melodyczno-rytmiczny wygenerowany przy użyciu programu opisanego w podrozdziale 4.1.2 na stronie 66.

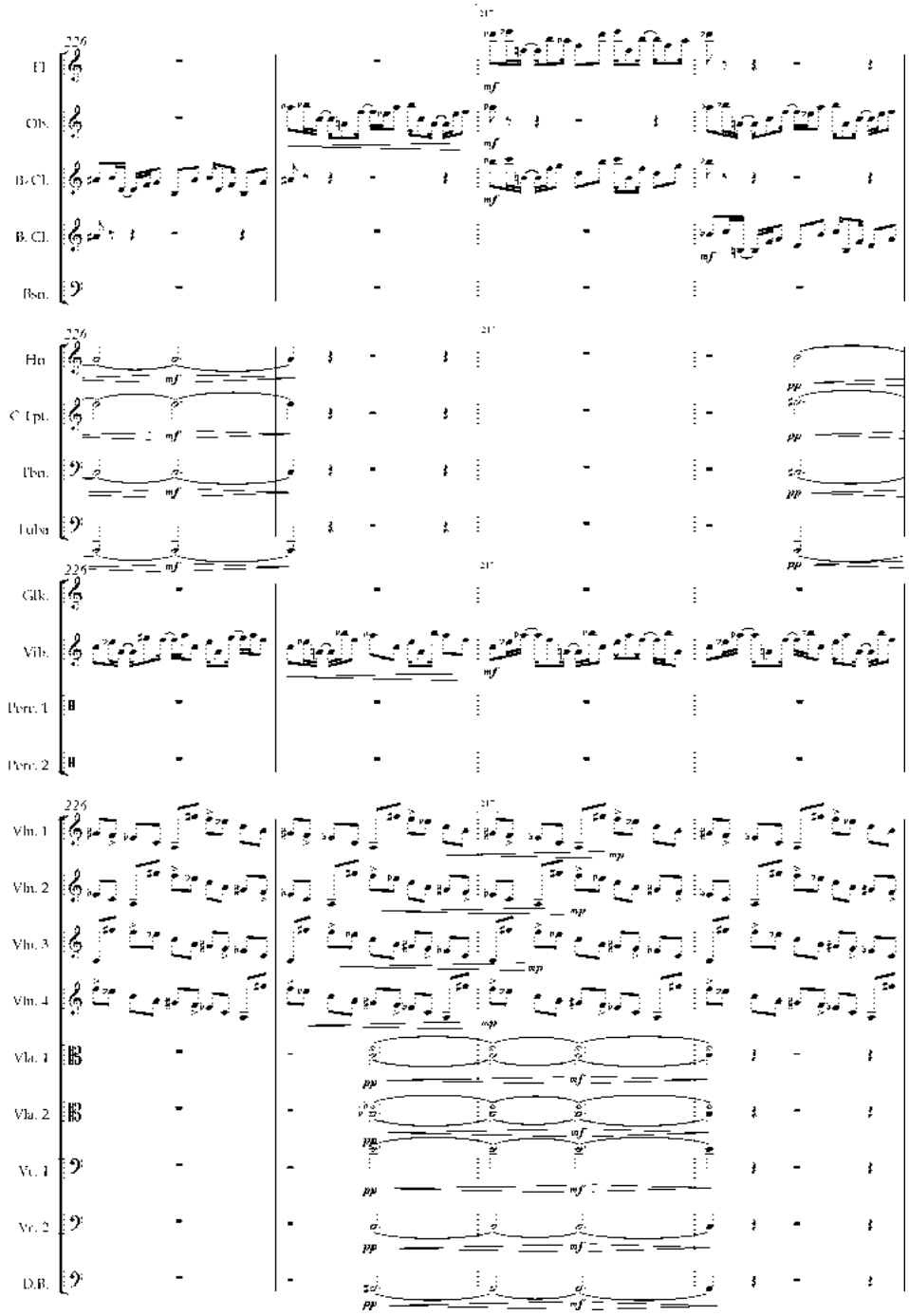


Rysunek 3.12: Progresja akordów wygenerowanych przez program pokazany na rysunku 4.12 ze strony 74.

Technika interpolacji pozwala nie tylko uzyskiwać nowy materiał dźwiękowy, lecz jest przede wszystkim ważną metodą kształtowania formalnego. Ewolucja materiału staje się tu bowiem procesem, którego rozwój wyznacza kierunek, kształt oraz logikę przebiegu formalnego w danym odcinku kompozycji. Spotykają się tu zatem dwie ważne kategorie: materiału i formy, które

[8]Interpolacja w matematyce, to wg *Słownika języka polskiego* [100]: "wyznaczenie w pewnym przedziale funkcji, która przyjmuje znane wartości dla danych liczb z tego przedziału."

[9]Podobne procedury stosowane są również w odniesieniu do rytmu. O różnych technikach interpolacji stosowanych przez kompozytorów szkoły spektralnej czytaj w: [43, 86].

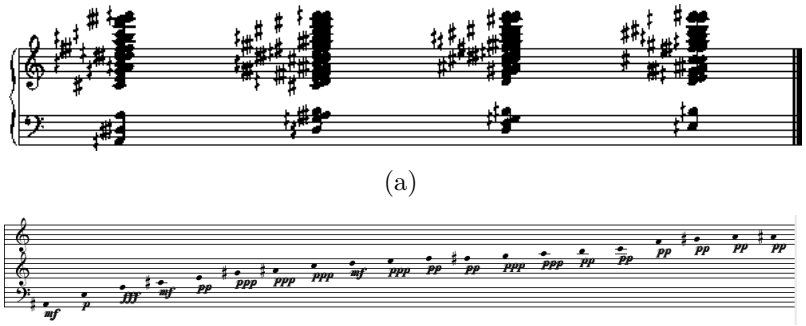


Rysunek 3.13: Fragment partytury (t. 226–229) drugiej części –*Moderato* utworu pt. *Rozbłyski*, przedstawiający sposób wykorzystania materiału dźwiękowego uzyskanego przy użyciu techniki interpolacji.

stają się jakby różnymi aspektami tego samego pojęcia lub dwiema stronami tej samej monety. Ich wzajemna interakcja nadaje zatem nowy sens tym, kluczowym w mówieniu o sztuce od czasów Arytotelesa, kategoriom[10].

3.3.6 Akordy "konkretne"

Przez akordy "konkretne" rozumie się materiał harmoniczny uzyskany w wyniku analizy nagrań zawierających próbki dźwięków istniejących w świecie realnym, tj. dźwięki instrumentów muzycznych, odgłosy natury, czy działalności człowieka. W przypadku utworu pt. *Rozbłyski* były to dwa dźwięki instrumentów perkusyjnych: tam-tamu oraz niskiego gongu. Próbki dźwiękowe zostały najpierw poddane analizie spektralnej, następnie zaś przefiltrowane w celu pozostawienia jedynie najgłośniejszych składników widma. Uzyskane w ten sposób informacje zostały następnie przekształcone do postaci zapisu nutowego, który oprócz wysokości dźwięków zawiera również dane na temat dynamiki poszczególnych składowych (patrz rysunek 3.14), długości ich trwania oraz kolejności występowania (rysunek 3.16). Dokładna procedura techniczna oraz programy umożliwiające uzyskanie użytecznego z punktu widzenia kompozycji instrumentalnej materiału dźwiękowego przedstawione są w rozdziale 4.2 na stronie 76.

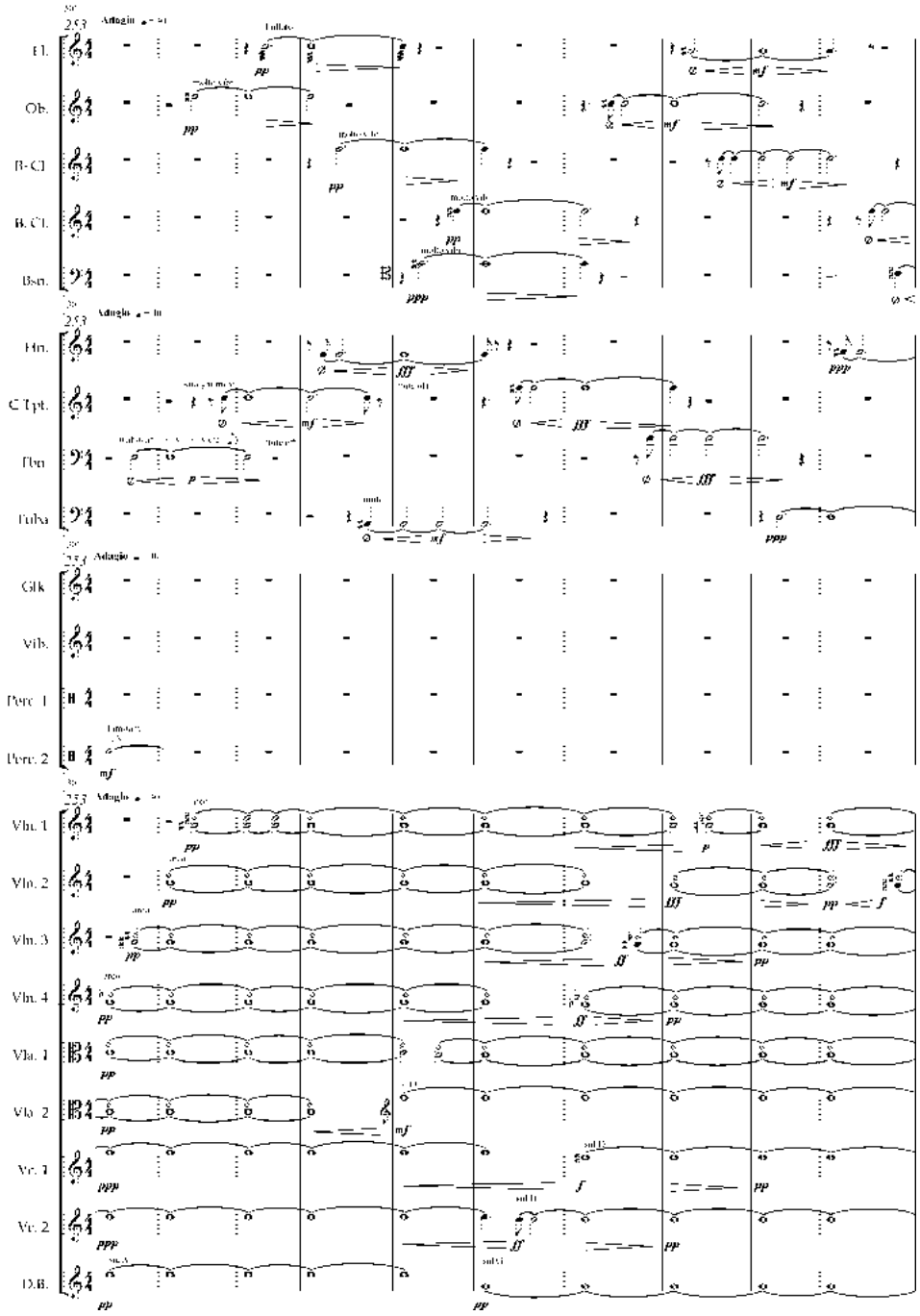


(a)

(b)

Rysunek 3.14: Analiza widma tam-tamu: (a) progresja zawierająca odczyty 4 faz rozwojowych dźwięku oraz (b) widok pierwszego akordu, zawierający dane na temat wysokości i dynamiki poszczególnych składowych.

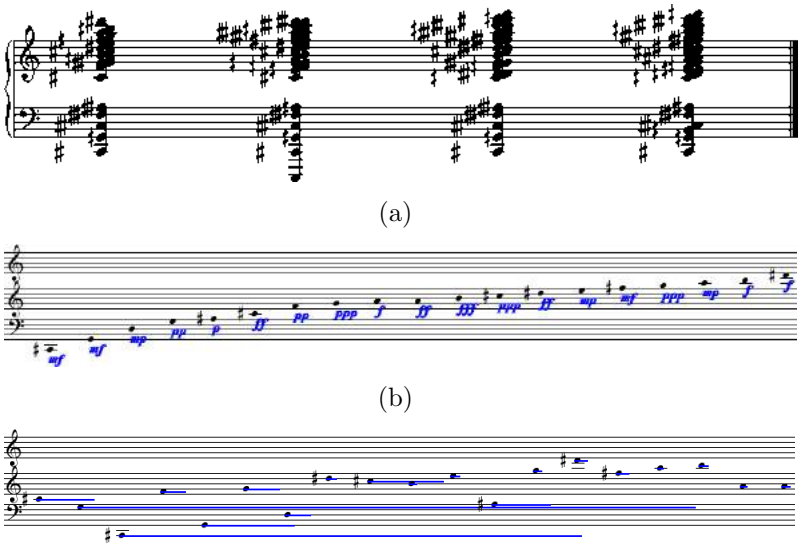
[10]O zagadnieniu relacji materii i formy w muzyce spektralnej czytaj w: [83].



Rysunek 3.15: Fragment partytury trzeciej części – *Adagio* utworu pt. *Rozbłyski*, przedstawiający sposób wykorzystania spektrum tam-tamu z rysunku 3.14.

Na rysunku 3.15 widzimy sposób wykorzystania materiału harmonicznego uzyskanego dzięki analizie widma tam-tamu. Spektrum to, o silnej charakterystyce szumowej zawierało kilkaset składowych, których liczba musiała zostać zredukowana w taki sposób, aby można było posługiwać się nimi w muzyce instrumentalnej (patrz rysunek 4.17 na stronie 78). Ponieważ spektra instrumentów naturalnych, w szczególności zaś idiofonów charakteryzują się dużą zmiennością w czasie, analizie poddano cztery różne fazy rozwoju dźwięku, uzyskując za każdym razem inny odczyt, co ukazuje rysunek 3.14. Przedstawiony tam materiał wykorzystano w trzeciej części – *Adagio* utworu pt. *Rozbłyski*, w taktach 253–271.

Podobną procedurę techniczną zastosowano w odniesieniu do próbki dźwiękowej niskiego gongu. Na rysunku 3.16 widzimy odczyty kolejnych faz rozwoju dźwięku oraz szczegóły pierwszego odczytu przedstawione w formie notacji muzycznej. Materiał ten użyty został w kulminacyjnym momencie utworu pt. *Rozbłyski*, czyli w trzeciej części – *Adagio. Doppio movimento*, w taktach 272–281. Przedstawione to zostało na rysunku 3.17.

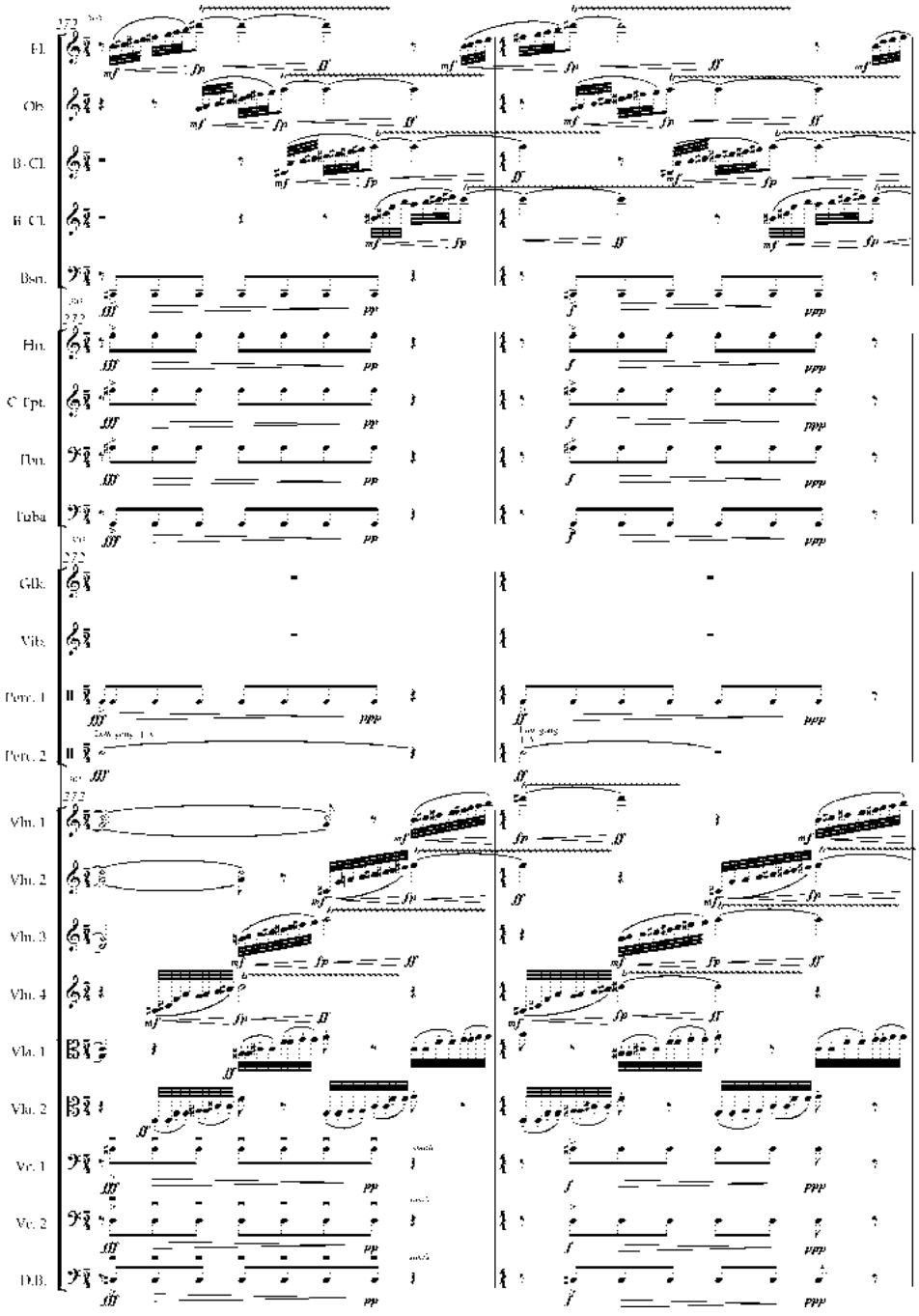


(a)

(b)

(c)

Rysunek 3.16: Analiza widma niskiego gongu: (a) progresja zawierająca odczyty 4 faz rozwojowych dźwięku, (b) odczyt przedstawiający dynamikę składowych oraz (c) odczyt przedstawiający długość i kolejność składowych.



Rysunek 3.17: Fragment partytury trzeciej części –*Adagio. Doppio movimento* utworu pt. *Rozbłyski*, przedstawiający sposób zastosowania materiału dźwiękowego uzyskanego z dźwięku niskiego gongu.

3.4 Forma oraz zasady kształtowania

Utwór pt. *Rozbłyski* zbudowany jest z czterech części:

I. Adagio assai (t. 1-79)

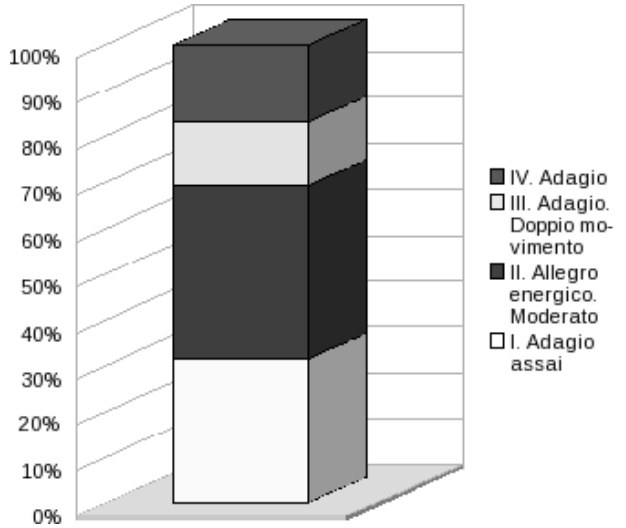
II. Allegro energico. Moderato (t. 80-252)

III. Adagio. Doppio movimento (t. 253-281)

IV. Adagio (t. 282-324)

Rysunek 3.18 przedstawia w postaci wykresu poszczególne części utworu. Reprezentowane są one w ujęciu procentowym wg czasu trwania. Tego typu prezentacja jest bliższa realnej, tzn. odbieranej przez słuchacza formie utworu. Należy jednakże mieć na uwadze fakt, iż czas psychologiczny związany jest przede wszystkim z ilością, częstotliwością oraz intensywnością docierających do nas impulsów, niekoniecznie więc pokrywa się rzeczywistą długością odcinków czasowych[11]. Trzeba przy tym zaznaczyć, że części: trzecia oraz czwarta wykonywane są bez przerw i wobec tego odbierane przez słuchaczy jako całość.

[11]Zjawisko to jest bez wątpienia dobrze znane z codziennych doświadczeń, gdy np. nudny mecz piłkarski "dłuży się", a mecz ciekawy kończy się "zbyt szybko".



Rysunek 3.18: Wykres ukazujący formę utworu pt. *Rozbłyski* w postaci procentowej.

Powodem wydzielenia w partyturze oraz na wykresie czwartej części, jako jednostki samodzielnej są względy koncepcyjne, tkwiące w zasadzie rządzącej rozwojem materiału. Forma utworu w swej najbardziej pierwotnej idei rozpięta jest bowiem pomiędzy dwiema skrajnościami: spektrum harmonicznym w pierwszej części oraz spektrum subharmonicznym w czwartej części. Te dwie antytetyczne kategorie brzmieniowe, odpowiadające w rozumieniu autora w sposób symboliczny pojęciom światła i mroku, wyznaczają skrajne punkty rozwoju formy, pomiędzy którymi rozpięta jest muzyka *Rozbłysków*.

Forma utworu kształtowana jest w sposób procesualny. Znaczy to, że naczelną kategorią formalną jest proces, rozumiany jako stopniowy rozwój materiału dźwiękowego. Tego typu podejście stanowi kontynuację postawy zarówno spektralistów, jak i minimalistów [92]. Kształtowanie formy jako procesu najwyraźniej widoczne jest w tych częściach utworu, w których wykorzystane zostały techniki: interpolacji (II. *Moderato*) i rotacji struktur harmonicznych (IV. *Adagio*), które to techniki opisane zostały w podrozdziale 3.3. Interesujący przykład zastosowania muzycznego procesu stanowi również środkowy fragment części drugiej (*Allegro energico*, t. 148-200), w którym materiał harmoniczny, zbudowany w oparciu o model syntezy FM został posortowany w taki sposób, iż słuchacz śledzi stopniową przemianę

kolorytu harmonicznego od ciemniejszego do jaśniejszego. Technika ta opisana została w podrozdziale 4.3.1 na stronie 82.

W celu lepszego zobrazowania zagadnienia formy utworu należy rozpatrzeć ją z kilku punktów widzenia: materiału, techniki, agogiki i energetyki. Poszczególne sfery tworzą złożoną sieć wzajemnych odniesień i zależności, co obrazuje tabela 3.1.

Nr	Sfera materiału	Sfera techniki	Sfera agogiki	Sfera energetyki
I.	Spektra harmoniczne	Repetycje, Permutacje rytmiczne	Adagio assai	Statyka
II.	Akordy FM	Ewolucjonizm Transformacja barwy Interpolacje harmoniczne Repetycje, Permutacje melodyczno-rytmiczne	Allegro energico Moderato	Dynamika Kulminacja Dynamika Kulminacja
III.	Akordy "konkretne"	Ciągłość Repetycje	Adagio Doppio movimento	Statyka Kulminacja (główna)
IV.	Szereg subharmoniczny i pochodne Materiał nieharmoniczny	Rotacja/Repetycje Rotacja/Ciągłość	Adagio	Statyka

Tablica 3.1: Różne aspekty formy utworu pt. *Rozbłyski*.

Rozdział 4

Zastosowania CAC w utworze pt. *Rozbłyski*

Systemy CAC w założeniu mają ułatwiać długi i często pracochłonny etap przygotowawczy, na który składają się m.in.: poszukiwanie, generowanie i weryfikacja materiału dźwiękowego. *OpenMusic*, dzięki swojej otwartości i modularności, dostarcza kompozytorowi nieograniczonych możliwości w zakresie projektowania własnych, odpowiadających indywidualnym potrzebom programów realizujących określone czynności kompozytorskie[1]. W trakcie pracy nad utworem pt. *Rozbłyski*, system ten wykorzystany został do tworzenia programów dostarczających różnorodnego materiału: rytmicznego, melodycznego oraz harmonicznego.

W dalszej części rozdziału przywołane zostaną przykłady zastosowania komputerowo wspomaganej kompozycji, w szczególności zaś wykorzystanie programu *OpenMusic*. Układ kolejnych podrozdziałów odpowiada rozmaitym czynnościom kompozytorskim, podejmowanym w procesie komponowania utworu pt. *Rozbłyski* z użyciem tego systemu. Składało się na nie:

[1]Funkcjonalność programu *OpenMusic* rozszerzają dodatkowo zestawy bibliotek, dostarczających narzędzi związanych m.in. z analizą i syntezą dźwięku, importowaniem i eksportowaniem plików w różnych formatach. Dają one także możliwość tworzenia muzyki algorytmicznej, m.in. w oparciu o modele Markowa, fraktale, czy znany jeszcze z *PatchWorku* model tzw. constraints programming, czyli poszukiwania rozwiązań w ramach wyznaczonych ram. Niestety nie wszystkie biblioteki udostępniane są w wersji bezpłatnej [1].

1. generowanie materiału dźwiękowego:
    a. generowanie przebiegów rytmicznych
    b. generowanie przebiegów melodyczno–rytmicznych
    c. generowanie progresji w oparciu o akordy stworzone na bazie spektrum harmonicznego
    d. generowanie akordów FM
    e. interpolacje harmoniczne
    f. inwersja i rotacja akordów
2. praca z materiałem "konkretnym". Składa się z dwóch etapów:
    a. analizowanie oraz filtrowanie nagranych próbek dźwiękowych
    b. przekształcanie uzyskanych rezultatów do postaci zapisu nutowego.
3. porządkowanie materiału harmonicznego:
    a. sortowanie akordów FM

4.1 Generowanie materiału dźwiękowego

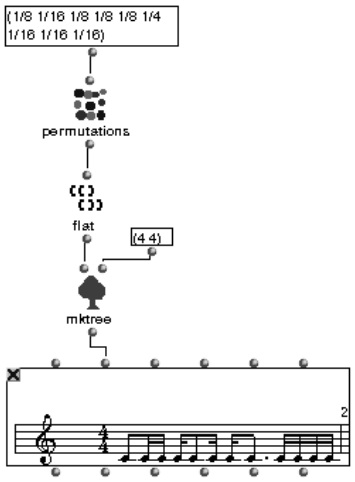
Zasadą tworzenia nowego materiału dźwiękowego było: (a) przekształcanie struktury istniejącej, (b) generowanie materiału na bazie określonej zasady formującej, tj. struktura spektrum harmonicznego, model syntezy FM, czy technika interpolacji.

4.1.1 Generowanie przebiegów rytmicznych

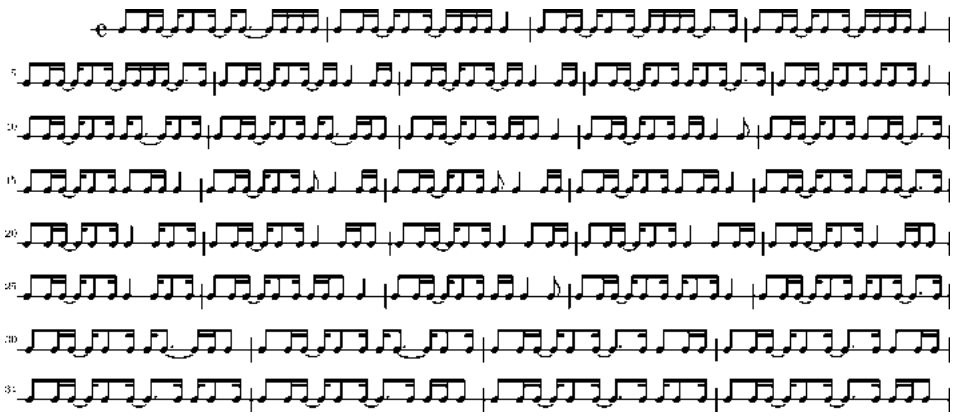
Przy użyciu *OpenMusic* stworzone zostały przebiegi rytmiczne oparte na permutacjach podstawowego modelu, pokazanego na rysunku 4.1:

Rysunek 4.1: Model rytmiczny nr 1.

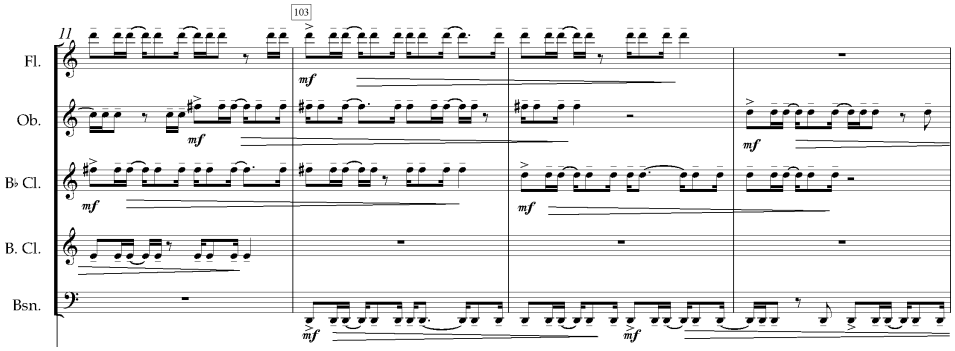
Program przedstawiony na rysunku 4.2 zrealizował permutację modelu, w wyniku której uzyskano bogaty materiał rytmiczny, obejmujący ponad siedemset wzorów (rysunek 4.3). Przebiegi rytmiczne dobierane były następnie w sposób swobodny i wykorzystane w partiach instrumentów dętych drewnianych w t. 4–38, 54–62 oraz 68–75, w I części – *Adagio assai* utworu pt. *Rozbłyski* (rysunek 4.4).



Rysunek 4.2: Widok łatki *OpenMusic* dokonującej permutacji modelu rytmicznego nr 1.



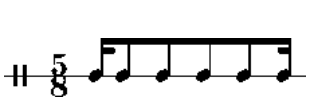
Rysunek 4.3: Fragment materiału rytmicznego wygenerowanego przy użyciu łatki z rysunku 4.2



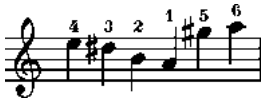
Rysunek 4.4: Fragment (t. 11–14) I części – *Adagio assai* utworu pt. *Rozbłyski*, ukazujący partie intrumentów dętych drewnianych, w których wykorzystano wzory rytmiczne z rysunku 4.3.

4.1.2 Generowanie przebiegów melodyczno–rytmicznych

W podobny sposób wytworzony został materiał melodyczno-rytmiczny, będący rezultatem przekształceń zarówno rytmu, jak i melodii. Rysunek 4.6 pokazuje program realizujący permutacje podstawowego modelu rytmicznego oraz melodycznego (rysunek 4.5).



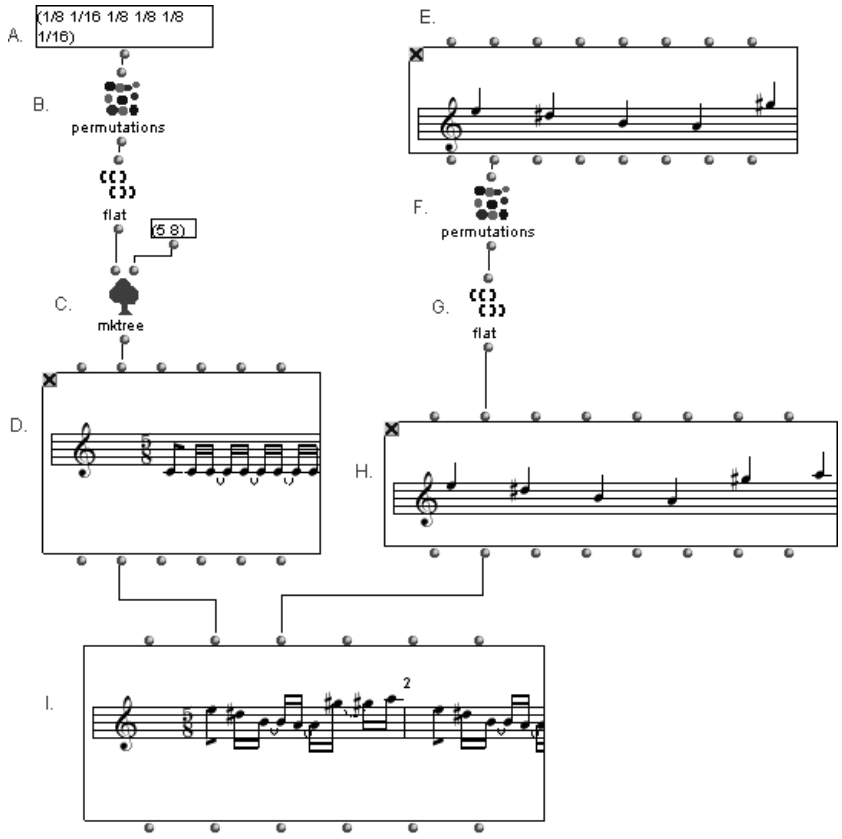
(a) Model rytmiczny nr 2



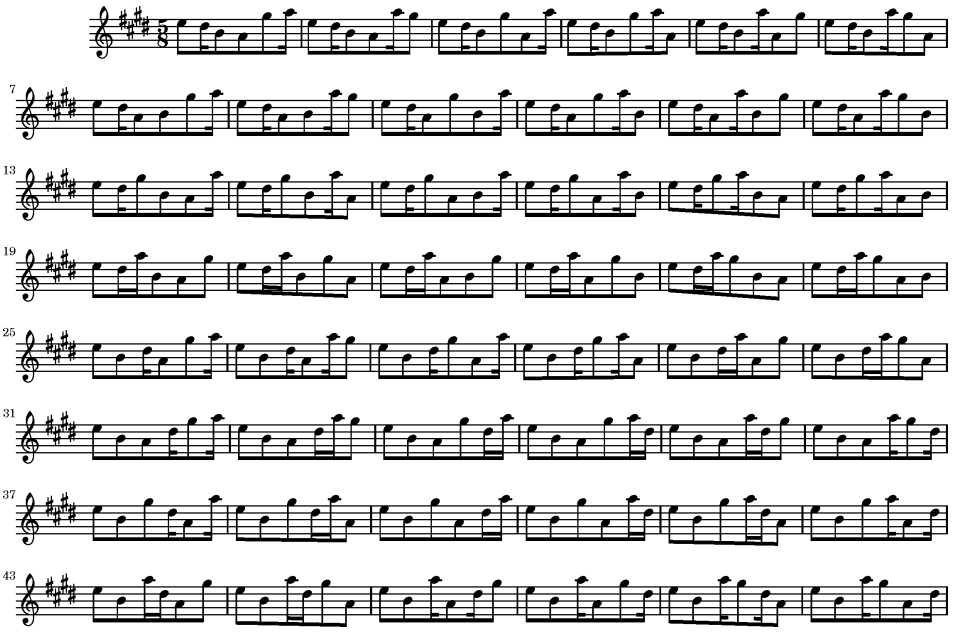
(b) Model melodyczny nr 1

Rysunek 4.5: Materiał wyjściowy dla permutacji melodyczno–rytmicznych.

Program przedstawiony na rysunku 4.6 wygenerował ponad siedemset wersji podstawowego materiału melodyczno-rytmicznego (rysunek 4.7). Stanowił on punkt wyjścia dla skomponowania partii wibrafonu oraz instrumentów dętych drewnianych w taktach 210–242, czyli we fragmencie II części - *Moderato*, utworu pt. *Rozbłyski*.



Rysunek 4.6: Widok łatki *OpenMusic* generującej przebiegi melodyczno–rytmiczne, wykorzystującej funkcję permutacji modeli wyjściowych.



Rysunek 4.7: Fragment materiału melodyczno-rytmicznego wygenerowanego przy użyciu programu przedstawionego na rysunku 4.6.

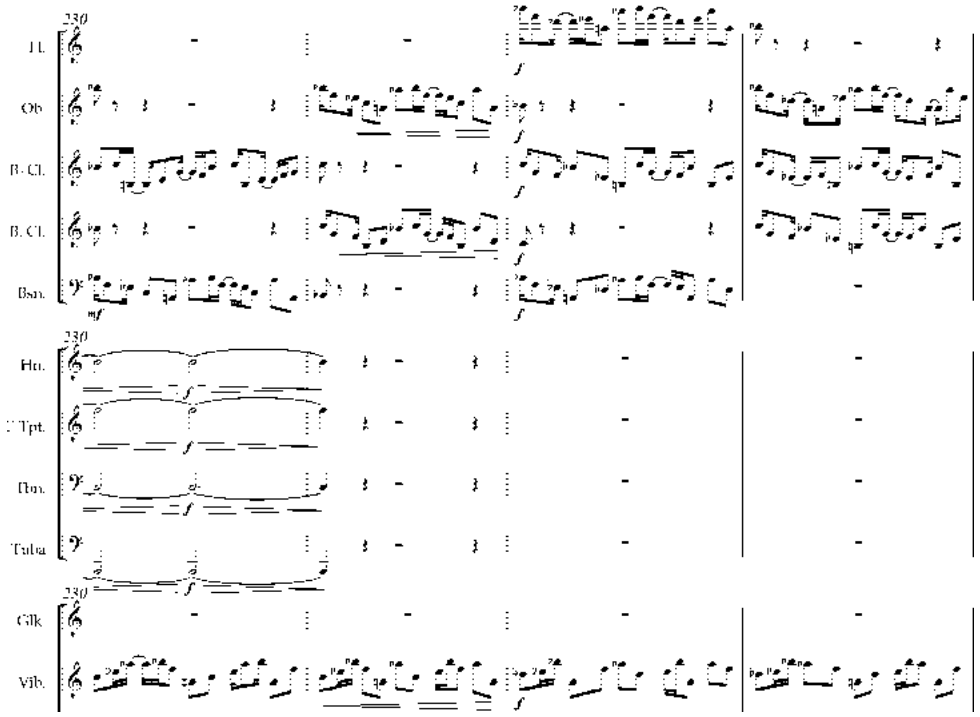
Wyjaśnienia wymaga sposób wykorzystania przytoczonego materiału. Odpowiednie modele wybierane były w sposób intuicyjny i każdorazowo dostosowywane do właściwego w danym fragmencie kontekstu harmonicznego. Odbywało się to w ten sposób, że poszczególnym wysokościom dźwięków modelu wyjściowego przypisano odpowiedni numer indeksu (patrz rysunek 4.5 b), który następnie dopasowywano do nowych ukształtowań dźwiękowych. Fragment przedstawiony na rysunku 4.8 wykorzystuje w partiach instrumentów dętych drewnianych oraz wibrafonu następujące układy dźwiękowe:

- W taktach 230–231:

$$a^1(1), c^2(2), es^2(3), f^2(4), as^2(5), b^2(6)$$

- w taktach 232–233:

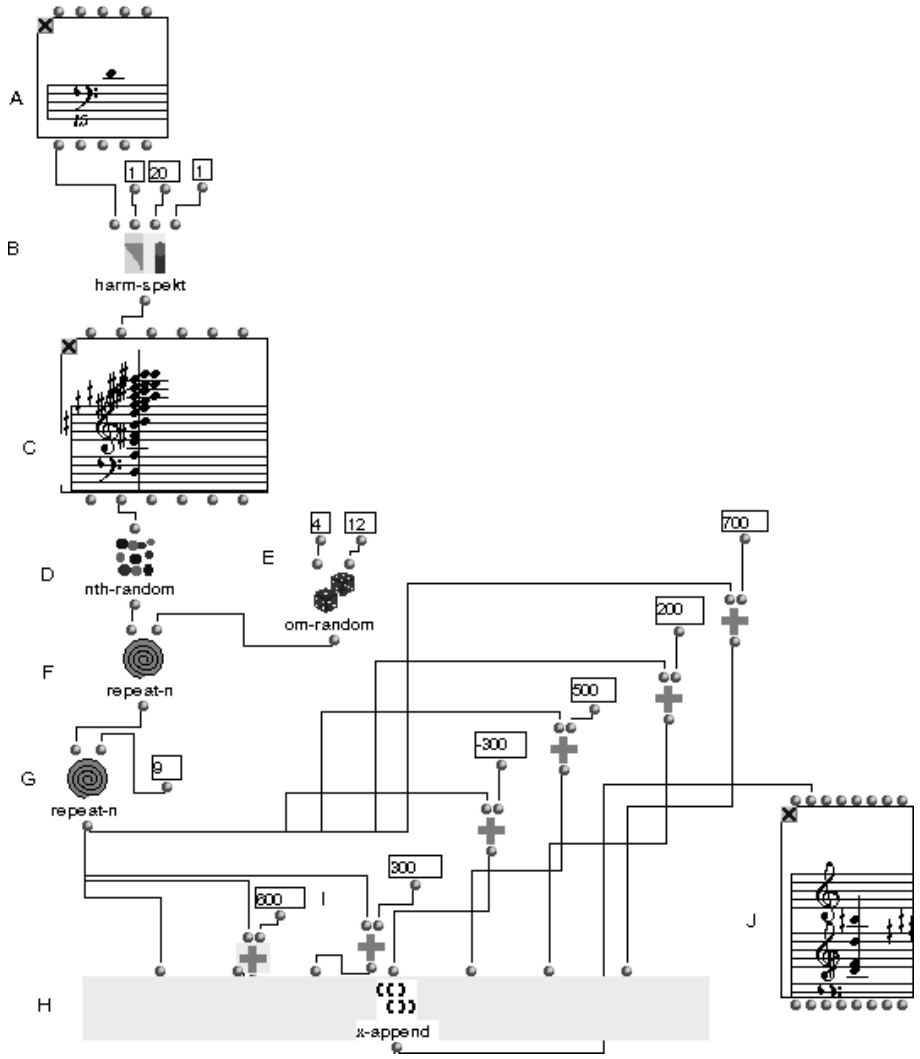
$$a^1(1), des^2(2), es^2(3), f^2(4), as^2(5), b^2(6).$$



Rysunek 4.8: Fragment partytury II części – *Moderato* utworu pt. *Rozbłyski*, w której wykorzystano materiał melodyczno–rytmiczny wygenerowany przy użyciu programu z rysunku 4.6.

4.1.3 Generowanie progresji w oparciu o akordy stworzone na bazie spektrum harmonicznego

OpenMusic jest użytecznym narzędziem do tworzenia różnego rodzaju materiału harmonicznego, a także wyznaczania progresji harmonicznych opartych na zasadach przedstawionych w formie algorytmicznej. Przy pomocy zdefiniowanej przez autora funkcji `harm-spekt` pokazanej na rysunku 4.9 można uzyskać akordy o różnej ilości oraz gęstości dźwięków, których układ odwzorowuje porządek istniejący w szeregu harmonicznym.



Rysunek 4.9: Widok łatki tworzącej progresję akordów zbudowanych na bazie spektrum harmonicznego, przedstawionej na rysunku 3.3 na stronie 42.

Program[2] przedstawiony na rysunku 4.9 oprócz tworzenia współbrzmień o strukturze szeregu harmonicznego dokonuje również ich "filtrowania" i transpozycji. Funkcja `x-append` generuje progresję harmoniczną, transponując akordy uzyskane przez `repeat-n` (moduł G). Ta ostatnia przy każdym wywołaniu zwraca dziewięć współbrzmień na bazie spektrum o różnej struk-

[2]Jest to zmodyfikowana wersja programu, którego autorem jest kompozytor Karim Haddad [11]. Wszystkie pozostałe programy przedstawione w tej pracy stworzone zostały przez jej autora.

turze i gęstości. Wyznaczenie tych ostatnich parametrów dokonywane jest z użyciem wyborów losowych, za które odpowiadają funkcje: `nth-random` (wybiera losowo składniki akordu przedstawionego w edytorze C) i `om-random` (losuje liczbę mieszczącą się w zakresie 4–12, wyznaczającą ilość dźwięków w akordzie). Rezultatem działania programu jest progresja harmoniczna przedstawiona na rysunku 3.3, znajdującym się na stronie 42.

4.1.4 Generowanie akordów FM

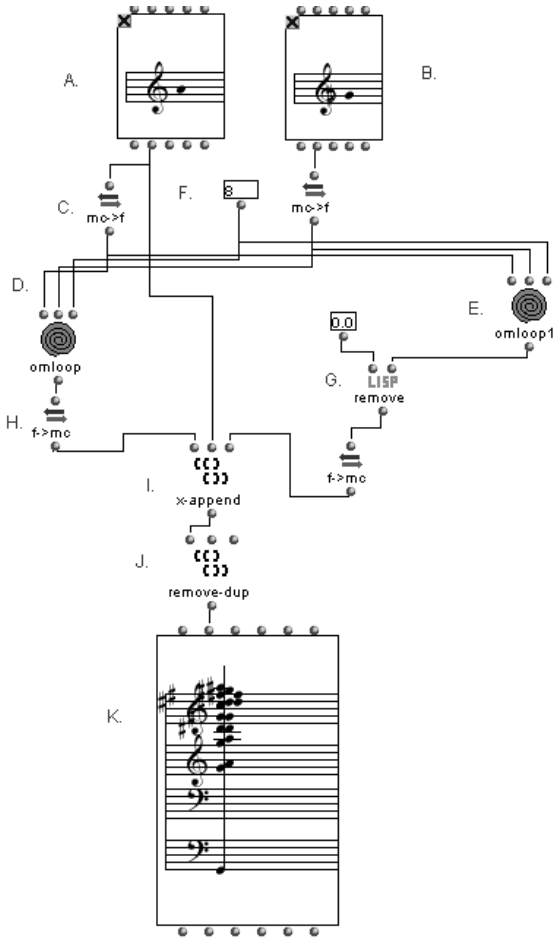
W podrozdziale 3.3.4 na stronie 49 przedstawiony został opis tzw. *akordów FM*, czyli struktur harmonicznych powstałych w oparciu o model syntezy przez modulację częstotliwości. W tym miejscu przedstawiony zostanie program generujący tego typu materiał harmoniczny. Jak powiedziano wcześniej, w syntezie FM w wyniku oddziaływania częstotliwością modulującą na częstotliwość nośną powstaje złożone spektrum dźwiękowe, które oprócz częstotliwości nośnej (f_c) zawiera także tzw. wstęgi boczne, różniące się od częstotliwości nośnej o kolejne wielokrotności częstotliwości modulującej (f_m) w górę i w dół (patrz rysunek 3.9 na stronie 50). Szereg taki można przedstawić za pomocą równania:

$$f(x) = f_c \pm (x * f_m)$$

gdzie x stanowi dowoną liczba całkowita w zakresie: $x = 1 \rightarrow \infty$. Przedstawiony tu sposób wyliczania akordów FM nie bierze pod uwagę amplitudy poszczególnych składowych. W metodzie Chowninga wartość ta, razem z wartością wskaźnika modulacji (ang.*modulation index*), wyrażana jest funkcją w czasie. Ponieważ w zastosowanej przez autora metodzie nie chodzi o cyfrową syntezę dźwięków, lecz wykorzystanie syntezy FM jako modelu dla układów harmonicznych, pierwsza z tych wartości została całkowicie pominięta, druga zaś traktowana jest jako maksymalna wartość x, podawana w obliczeniu jako stała (patrz moduł F na rysunku 4.10).

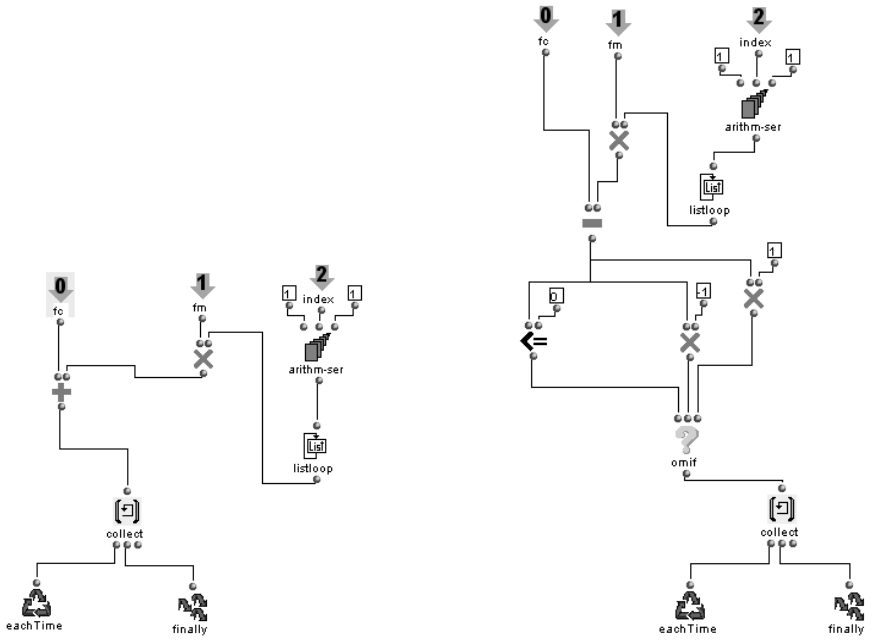
Rysunki 4.10 oraz 4.11 przedstawiają program generujący akordy FM, wykorzystane w drugiej części utworu *Rozbłyski –Allegro energico. Moderato.* Dokładny opis struktur harmonicznych, uzyskanych przy użyciu przed-

stawionego tu programu oraz sposób ich wykorzystania znajduje się w podrozdziale 3.3.4 na stronie 49.



Rysunek 4.10: Widok łatki generującej akordy w oparciu o model syntezy FM.

Program widoczny na rysunku 4.10 ukazuje: częstotliwość nośną (moduł A), częstotliwość modulującą (moduł B) oraz rezultat modulacji w postaci akordu widocznego u dołu (moduł K). Z kolei pętle, w postaci łat przedstawionych na rysunku 4.11, realizują szereg operacji dodawania (`omloop` – moduł D) oraz oraz odejmowania (`omloop1` – moduł E) kolejnych wielokrotności częstotliwości modulującej do/od częstotliwości nośnej. Ilość iteracji w pętli, a tym samym ilość dźwięków w akordzie, uzależniona jest od wartości w polu F, która odpowiada wskaźnikowi modulacji.



(a) (b)

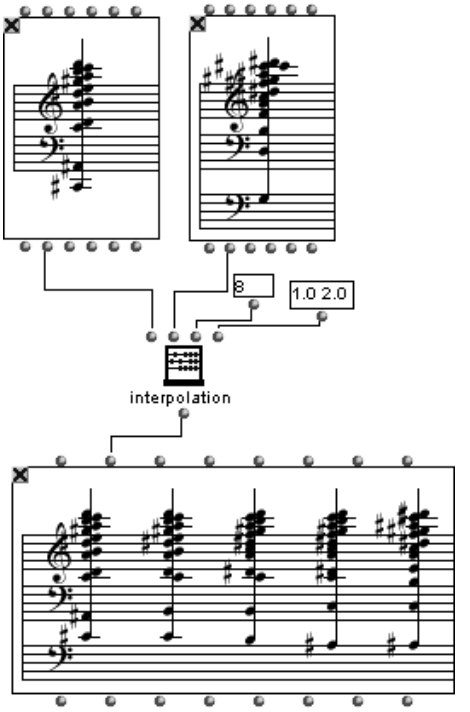
Rysunek 4.11: (a) Widok pętli D (`omloop`) oraz (b) pętli E (`omloop1`) z rysunku 4.10.

4.1.5 Generowanie progresji harmonicznych w oparciu o technikę interpolacji

Technika interpolacji oraz jej zastosowanie w odniesieniu do struktur harmonicznych, opisane zostało w podrozdziale 3.3.5 na stronie 53. Wyliczenia wartości poszczególnych węzłów funkcji, odpowiadających kolejnym wygenerowanym akordom, dokonuje funkcja `interpolation` widoczna na rysunku 4.12. Pobiera ona wartości struktury początkowej (akord FM widoczny z lewej górnej strony) i wartości struktury końcowej (akord FM widoczny z prawej górnej strony). Następnie funkcja ta wylicza stopnie pośrednie według podanej liczby kroków (jest ich 8) i charakterystyki krzywej interpolacji. W tym przypadku użyto funkcji wklęsłej[3], o czym decydują parametry `1.0 2.0`,

[3]"Rozkład (wartości) w funkcji wklęsłej wydaje się najlepiej odwzorowywać występowanie zjawisk naturalnych w naturze."[69, s. 65]

widoczne z prawej strony. Wynik działania programu stanowi progresja harmoniczna ukazana na rysunku 3.12 ze strony 53.

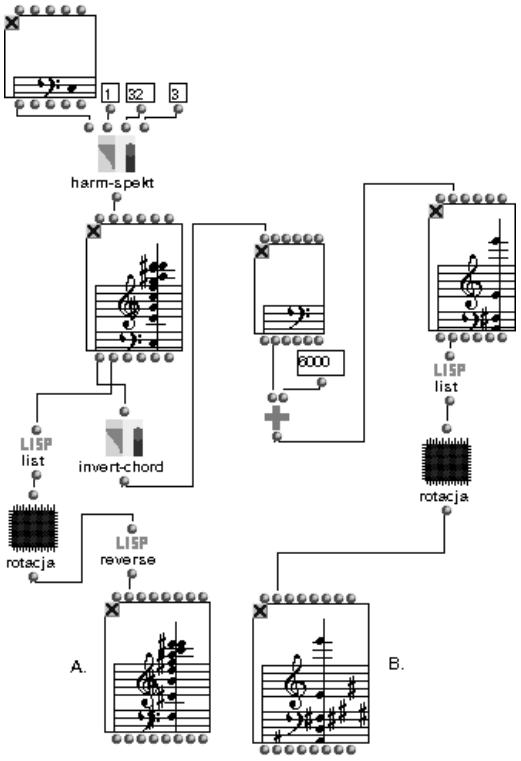


Rysunek 4.12: Widok łaty dokonującej interpolacji pomiędzy dwoma akordami FM.

4.1.6 Inwersja i rotacja akordów

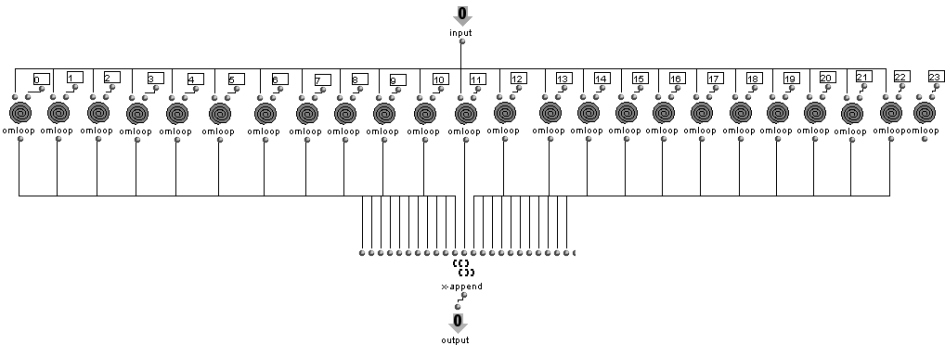
Technika rotacji oraz materiał harmoniczny powstały w wyniku jej użycia opisane zostały w podrozdziale 3.3.3 na stronie 45. Na rysunku: 4.13, przedstawiony został program dokonujący rotacji szeregów: harmonicznego i subharmonicznego. Widzimy na nim funkcję `harm-spekt`, która generuje spektrum harmoniczne dźwięku E_1, zawierające składowe od 1 do 32. Spektrum to jest dodatkowo przefiltrowane, tzn. zawiera jedynie co trzecią harmoniczną. Zdefiniowana przez autora funkcja `invert-chord` dokonuje inwersji spektrum harmonicznego. Po przetransponowaniu rezultatu działania tej funkcji uzyskano przefiltrowany szereg subharmoniczny dźwięku e^3. Struktury te zostały następnie poddane rotacji. Jej rezultaty widzimy na tym samym ry-

sunku: u dołu z lewej (A) dla spektrum harmonicznego, u dołu z prawej (B) dla struktury subharmonicznej.

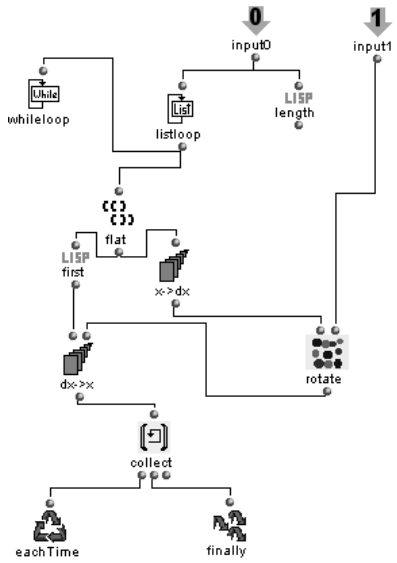


Rysunek 4.13: Widok łatki dokonującej rotacji akordów.

Podprogram `rotacja` widoczny na rysunku 4.14 składa się z szeregu takich samych pętli. Każda z nich dokonuje pojedynczego przesunięcia wejściowej struktury, którą jest szereg harmoniczny lub subharmoniczny o podaną z prawej strony liczbę składników. Wnętrze przykładowej pętli przedstawione jest na rysunku 4.15.



Rysunek 4.14: Widok wnętrza podprogramu `rotacja` z rysunku 4.13.



Rysunek 4.15: Widok wnętrza jednej z pętli `omloop` z rysunku 4.14.

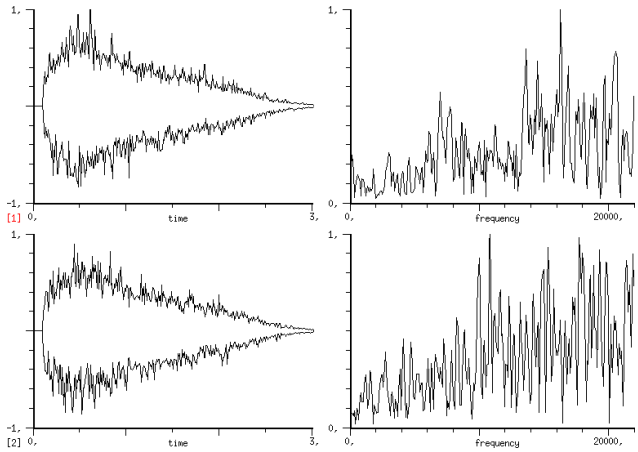
4.2 Praca z materiałem "konkretnym"

Praca z materiałem "konkretnym", oprócz samego nagrania obejmuje dwa etapy:

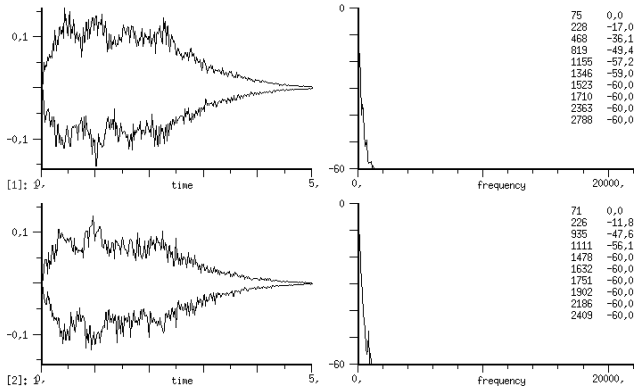
1. Analizowanie próbek dźwiękowych.
2. Przekształcanie uzyskanych rezultatów do postaci zapisu nutowego, którym można posługiwać się na dalszych etapach pracy nad utworem.

4.2.1 Analizowanie materiału "konkretnego"

W utworze "Rozbłyski" wykorzystane zostały struktury harmoniczne uzyskane z dwóch próbek realnych dźwięków instrumentów muzycznych: tam-tamu (rysunek 4.16 a) oraz niskiego gongu (rysunek 4.16 b). Opis tych struktur oraz sposobu ich wykorzystania znajduje się w podrozdziale 3.3.6 na stronie 55. W tym miejscu przedstawiona zostanie procedura techniczna prowadząca do pozyskania z próbek dźwiękowych użytecznego, z punktu widzenia kompozytora materiału harmonicznego.



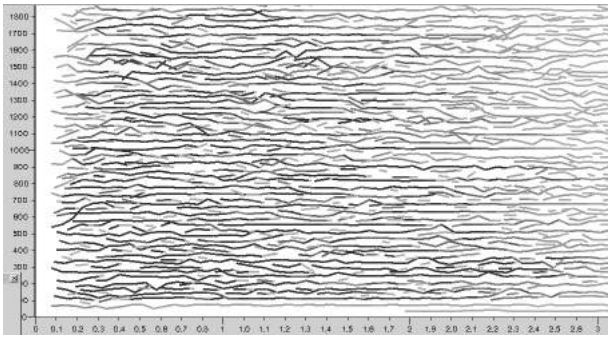
(a) Dźwięk tam-tamu



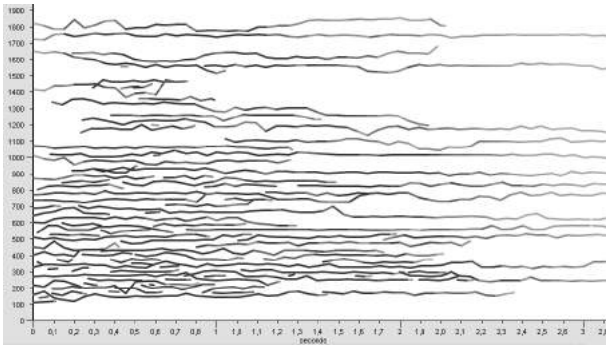
(b) Dźwięk niskiego gongu

Rysunek 4.16: Widok próbek dźwiękowych wykorzystanych do generowania materiału harmonicznego.

Na tym etapie pracy wykorzystano program *SPEAR*. Przedstawia on rezultat analizy spektralnej dźwięku w formie dwuwymiarowego wykresu zawierającego poszczególne składowe odwzorowane na osi rzędnych (w hercach) oraz ich rozwój w czasie przedstawiony na osi odciętych (w sekundach). Dźwięki instrumentów, które zostały poddane analizie (tam-tam i gong) posiadają bardzo bogate spektrum harmoniczne o charakterystyce szumowej, z ponad setką składowych nieharmonicznych. Tak wielka ilość materiału dźwiękowego była z punktu widzenia potrzeb muzyki instrumentalnej nieporządana, w związku z tym wstępny rezultat analizy został przefiltrowany w celu uzyskania najistotniejszych składowych. Odrzucone zostały najkrótsze (trwające poniżej 0.1 sek.) oraz najcichsze (poniżej -30 dB) z nich (rysunek 4.17).



(a) Widmo dźwięku tam-tamu, zawierające wszystkie składowe.

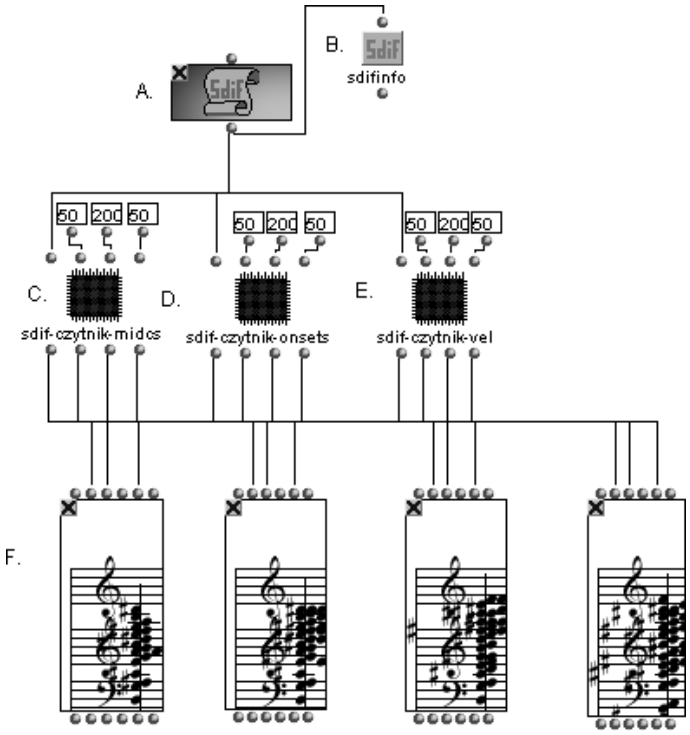


(b) Przefiltrowane widmo dźwięku tam-tamu, zawierające tylko wybrane składowe.

Rysunek 4.17: Widok analizy spektralnej dźwięku tam-tamu w oknie programu *SPEAR*.

4.2.2 Przekształcanie rezultatów analizy spektralnej do postaci notacji muzycznej

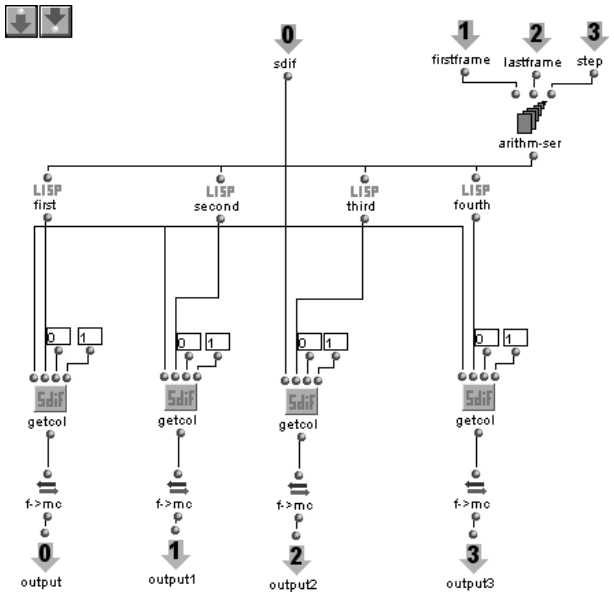
Kolejny etap składał się z wyeksportowania rezultatów analizy i filtracji spektrum z programu *SPEAR* do programu *OpenMusic*. Współdziałanie obydwu aplikacji umożliwia format SDIF (Sound Description Interchange Format)[90, 91]. Łata przedstawiona na rysunku 4.18 "wycina" z pliku SDIF pojedyncze ramki zawierające informacje o występujących w danej chwili składowych: ich ilości, częstotliwości, długości i dynamice oraz zamienia te informacje do postaci zapisu nutowego. Rezultatem działania programu są cztery różne akordy odwzorowujące różne momenty rozwoju analizowanego dźwięku. Przedstawiają one w formie notacji muzycznej informacje zawarte w ramkach nr: 0, 50, 100 i 150, czyli w początkowej, środkowej i końcowej fazie wybrzmiewania, w przypadku tam-tamu oraz w ramkach nr: 50, 100, 150 i 200, czyli nieznacznie opóźnionych w przypadku gongu.



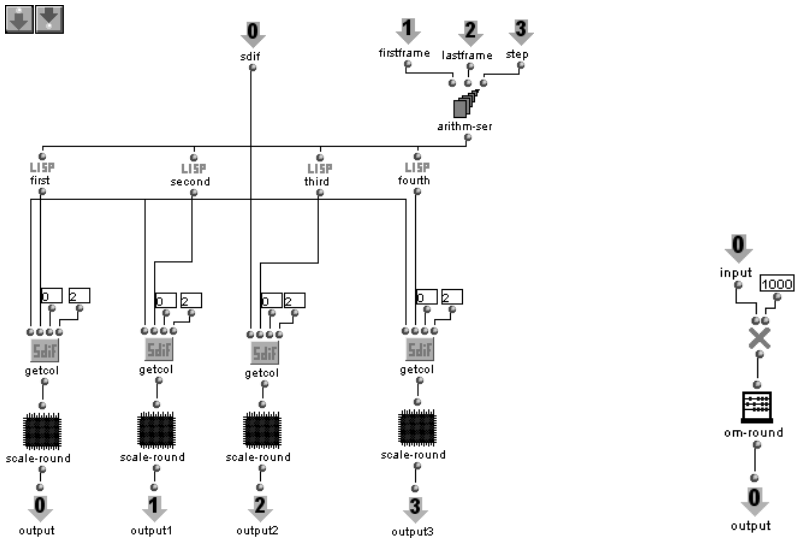
Rysunek 4.18: Widok łatki *OpenMusic* przekształcającej wyniki analizy spektralnej dźwięku do postaci notacji nutowej.

Program widoczny na rysunku 4.18 składa się w rzeczywistości z trzech podprogramów, z których każdy ma za zadanie odczytać informacje dotyczące określonego parametru analizowanego dźwięku:

- `sdif-czytnik-midics` odczytuje oraz konwertuje do wartości `midi-cents` (sposób reprezentacji wysokości dźwięków w programie *OpenMusic*) informacje dotyczące częstotliwości poszczególnych składowych (rysunek 4.19)
- `sdif-czytnik-onsets` odczytuje oraz konwertuje do wartości parametru `onsets`, czas wystąpienia oraz długości trwania poszczególnych składowych. Ponieważ skala analizowanych zjawisk jest niezwykle mała, w podprogramie `scale-round` zostają one przeskalowane, przy wartości referencyjnej odpowiadającej 1 sekundzie (rysunek 4.20)
- `sdif-czytnik-vel` odczytuje informacje na temat głośności poszczególnych składowych. Z pomocą podprogramu `db-vel` dokonuje się skalowania oraz przekonwertowania wartości ujętych w decybelach do wartości parametru `velocity` (rysunek 4.21)

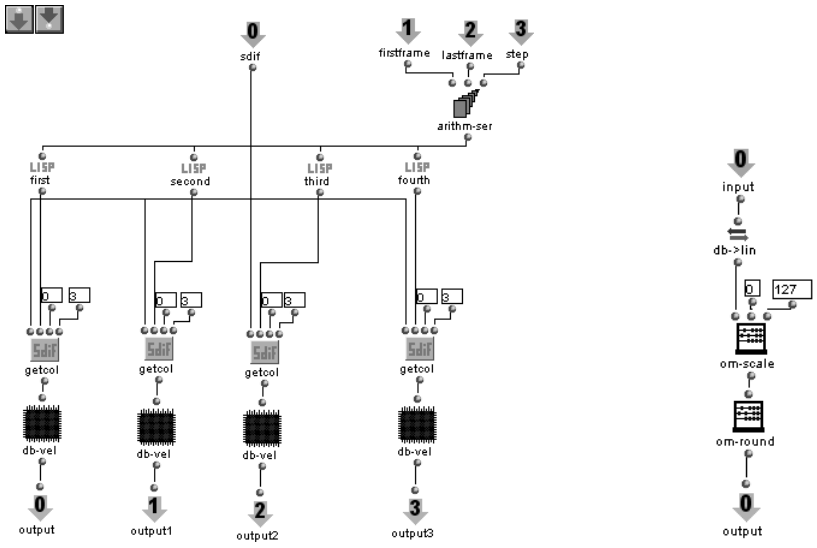


Rysunek 4.19: Widok wnętrza podprogramu `sdif-czytnik-midics` z rysunku 4.18.



(a) (b)

Rysunek 4.20: (a) Widok wnętrza podprogramu `sdif-czytnik-onsets` z rysunku 4.18 i (b) wnętrze podprogramu `scale-round`.



(a) (b)

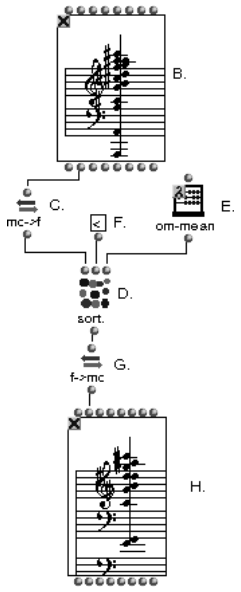
Rysunek 4.21: (a) Widok wnętrza podprogramu `sdif-czytnik-vel` z rysunku 4.18 i (b) wnętrzne podprogramu `db-vel`.

4.3 Porządkowanie materiału dźwiękowego

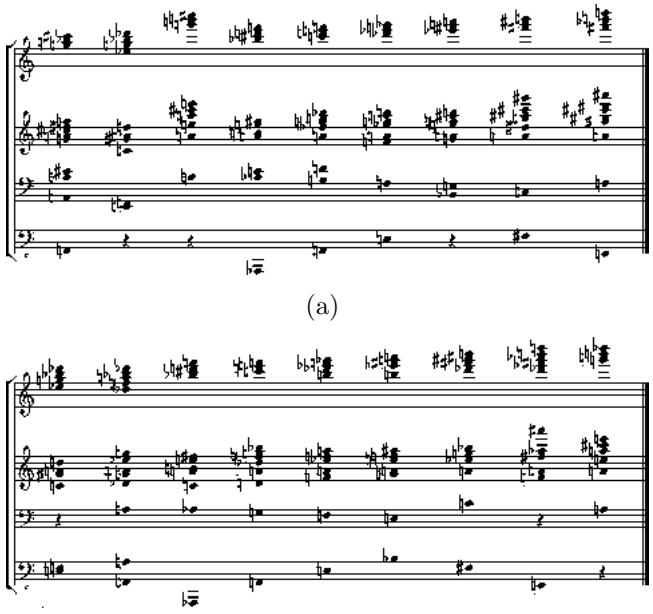
W kolejnym podrozdziale przedstawiony zostanie inny jeszcze sposób użycia narzędzi CAC. Polega on na zastosowaniu określonej procedury algorytmicznej w celu uporządkowania wygenerowanego uprzednio materiału dźwiękowego zgodnie z przyjętą zasadą.

4.3.1 Sortowanie akordów FM

Pod pojęciem sortowania akordów rozumie się wyznaczanie kolejności akordów w progresji harmonicznej. Możliwe jest przyjęcie rozmaitych kryteriów służących porządkowaniu materiału dźwiękowego. W tym przypadku sortowanie odbyło się na podstawie parametru, który możemy nazwać *barwą harmoniczną* lub kolorytem brzmieniowym: akordy FM uporządkowane zostały od "najciemniejszego" do "najjaśniejszego". Klasyfikacja akordów dokonana została na podstawie obliczeń wartości w aspekcie częstotliwościowym, wszystkich dźwięków składających się na dane współbrzmienie. Według tej klasyfikacji akordy, w których częstotliwości dźwięków składowych dają w sumie mniejszą wartość są traktowane jako "ciemniejsze", podobnie jak dźwięk a o częstotliwości 220Hz jest postrzegany jako "ciemniejszy" od a^1 o częstotliwości 440Hz. O ile w przypadku pojedynczych dźwięków, czy prostych współbrzmień metoda ta wydaje się być trywialna, o tyle zastosowanie jej do złożonych wielodźwiękowych struktur o bardzo zróżnicowanej budowie pozwala uzyskać ciekawe rezultaty, które widzimy na rysunku 4.23. Przedstawiona tam progresja harmoniczna została wykorzystana w taktach 148-200, czyli w drugiej części–*Allegro energico. Moderato* utworu pt. *Rozbłyski*.



Rysunek 4.22: Widok łaty sortującej akordy FM według ich kolorytu brzmieniowego, czyli wartości w domenie częstotliwości.



(a)

(b)

Rysunek 4.23: Widok progresji akordów FM: (a) nieposortowanych, (b) posortowanych przy użyciu programu przedstawionego na rysunku 4.22.

Rozdział 5

Podsumowanie

W opublikowanym po raz pierwszy w 1984 roku artykule zatytułowanym "Spectra and Sprites" [75] Tristan Murail pisał o pilnej potrzebie stworzenia systemu komputerowo wspomaganej kompozycji (CAC), jako interaktywnego środowiska zbudowanego na wzór systemów komputerowo wspomaganego projektowania (CAD). Jego celem miało być służenie kompozytorom w kreowaniu nowych, nieznanych wcześniej światów dźwiękowych. Gwałtowny rozwój technologii oraz postępy na polu programowania komputerów sprawiły, że w ciągu ostatnich kilkunastu lat powstało kilka tego typu systemów.

Niniejsza praca przynosi opis zastosowania jednego z nich, *OpenMusic*, w procesie komponowania utworu pt. *Rozbłyski*. Zaprezentowany tu materiał analityczny umożliwia czytelnikowi zapoznanie się zarówno z głównymi elementami języka dźwiękowego użytego w tym utworze, zastosowanymi tu technikami kompozytorskimi, jak i stworzonymi specjalnie dla potrzeb tej kompozycji, narzędziami CAC w postaci napisanych w środowisku *OpenMusic* programów. Zamierzeniem autora było takie ukazanie tematu, aby każde z głównych zagadnień niniejszej pracy zostało omówione w sposób dogłębny, a jednocześnie przejrzysty. Użyty tu system odnośników, odsyłających czytelnika do innych rozdziałów pracy lub odpowiednich fragmentów partytury, daje możliwość pełnego zrozumienia wzajemnych relacji występujących pomiędzy językiem kompozytorskim, a narzędziami komputerowo wspomaganej kompozycji. Jest to stosunek, który można nazwać *komplementarnością*,

a który cechuje wzajemne dopełnianie się idei i techniki kompozytorskiej. Zastosowanie komputera bowiem nie tylko ułatwiło i przyśpieszyło proces powstawania utworu, ale przede wszystkim umożliwiło urzeczywistnienie niektórych koncepcji kompozytorskich, które były niemożliwe do zrealizowania bez pomocy narzędzi informatycznych.

Podsumowując zakres zastosowania CAC w procesie twórczym prowadzącym do powstania utworu pt. *Rozbłyski*, należy przypomnieć, iż obejmował on m.in.:

- generowanie materiału dźwiękowego, przez zastosowanie permutacyjnych przekształceń
- tworzenie progresji akordów, powstających na bazie wygenerowanego materiału. Układy te, będące jednostkami wyższego rzędu, stanowią realizację określonych zasad formujących, tj.: transpozycja, interpolacja harmoniczna, rotacja.
- analizę, filtrowanie i konwertowanie materiału "konkretnego" do postaci zapisu nutowego. Dało to kompozytorowi dostęp do niezmierzonego bogactwa dźwięków naturalnych, mogących z powodzeniem służyć jako model dla kompozycji instrumentalnej.
- porządkowanie według przyjętej zasady wygenerowanego wcześniej materiału (sortowanie)

Przytoczone tu przykłady stanowią więc stosunkowo szeroki wachlarz procedur technicznych. Wobec powyższych faktów wydaje się być uprawnionym stwierdzenie, że wykorzystanie komputerowo wspomaganej kompozycji wpłynęło na wiele aspektów języka dźwiękowego i techniki kompozytorskiej autora. Wymienione i opisane w niniejszej pracy czynności techniczne, związane z realizacją procesu twórczego, wymagają wielkiej ilości czasochłonnych obliczeń, dokonywania żmudnego i skomplikowanego przekształcania danych. W związku z tym trzeba podkreślić, że jakkolwiek w pracy kompozytora nieodłącznym elementem była zawsze i zawsze pozostanie "idea artystyczna", to możliwość posłużenia się najnowocześniejszymi narzędziami komputerowymi daje twórcy szansę przeniesienia się na inny, wyższy poziom techniczny.

Rozszerza to pole kreacji artystycznej, daje możliwość lepszego, pełniejszego wypowiedzenia się. Zawsze powinno jednakże pozostawać w służbie sztuki, którą autor rozumie jako dialog człowieka z drugim człowiekiem.

Rozdział 6

Streszenia i słowa kluczowe

6.1 Streszczenie

Głównym tematem niniejszej pracy jest przedstawienie utworu orkiestrowego pt. *Rozbłyski* jako przykładu zastosowania komputerowo wspomaganej kompozycji. CAC jest dziedziną informatyki przeżywającą w ostatnich kilkunastu latach dynamiczny rozwój. Jej rosnąca popularność wśród kompozytorów o różnej orientacji stylistycznej rozpoczęła się w początkach lat dziewięćdziesiątych XX wieku i związana jest z jednej strony z upowszechnieniem się komputerów osobistych, z drugiej zaś z zastosowaniem nowego paradygmatu programowania wizualnego. Jednym z pierwszych systemów CAC, stosowanym powszechnie przez czołowych kompozytorów europejskich, tj. Brian Ferneyhough, Gérard Grisey, Paavo Heininen, Magnus Lindberg, Tristan Murail, czy Kaija Saariaho, był stworzony w paryskim IRCAM-ie *PatchWork*. Najnowocześniejszym obecnie stosowanym narzędziem CAC jest natomiast spadkobierca *PatchWorku* - *OpenMusic*. Praca przedstawia rozwój oraz stosunkowo szeroki kontekst zarówno historyczny, jaki i współczesny w jakim funkcjonuje dzisiejsza CAC. Główna jej cześć analizuje te elementy języka kompozytorskiego autora, których kształt zdefiniowany został przez zastosowanie narzędzi informatycznych. Analizy te uzupełnione są przedstawieniem i szczegółowym wyjaśnieniem użytych w procesie komponowania utworu pt. *Rozbłyski*, specjalnie w tym celu stworzonych w środowisku *OpenMusic*, pro-

gramów. Narzędzia te wykorzystano m.in. do generowania rytmów, melodii, akordów FM, czy progresji harmonicznych opartych na technikach tj.: filtrowanie i analizowanie spektrów, interpolacja harmoniczna i rotacja.

Słowa kluczowe: komputerowo wspomagana kompozycja (CAC), kompozycja algorytmiczna, muzyka komputerowa, muzyka spektralna, graficzne języki programowania, OpenMusic.

6.2 Abstract

This thesis deals with an orchestral composition entitled *Rozbłyski*, which serves as an example of a piece of music created with Computer Aided Composition system. Although CAC is a relatively new area of Information Technology, it has seen a rapid development in recent years. Its growing popularity among composers representing different stylistic orientations dates back to the early 1990s and is a result of a more widespread use of personal computers as well as the introduction of a new paradigm of visual programming. One of the first CAC systems to be commonly used by leading European composers i.e. Brian Ferneyhough, Gérard Grisey, Paavo Heininen, Magnus Lindberg, Tristan Murail or Kaija Saariaho was *PatchWork*, created at IRCAM in Paris. It was followed up by *OpenMusic*, which has come to be widely regarded as the most state-of-the-art CAC tool available nowadays. This thesis presents a development of CAC as well as a relatively extensive historical background of its use up to the present day. The main part of the thesis analyses the elements of the author's compositional language which were determined by the use of IT tools. Furthermore, the thesis presents and thoroughly describes *OpenMusic* patches which were specifically created to be used in the composition of *Rozbłyski*. The patches were used to generate rhythms, melodies, FM chords, or harmonic progressions based on such techniques as spectres filtering and analysis, harmonic interpolation and rotation.

tłum. Marcin Kilarski

Keywords: Computer Aided Composition (CAC), algorithmic composition, computer music, spectral music, visual programming languages, OpenMusic.

Bibliografia

[1] OpenMusic. Strona domowa projektu. Online. http://recherche.ircam.fr/equipes/repmus/OpenMusic/.

[2] Gottfried Michael Koenig. Online, 2000. http://www.koenigproject.nl, dostęp: 2009-12-05.

[3] Ada. Online, 2009. http://pl.wikipedia.org/wiki/Ada_(informatyka), dostęp: 2009-11-20.

[4] Ada Lovelace. Online, 2009. http://pl.wikipedia.org/wiki/Ada_Lovelace, dostęp: 2009-11-20.

[5] Komputerowe wspomaganie projektowania. Online, 2009. http://pl.wikipedia.org/wiki/Komputerowe_wspomaganie_projektowania, dostęp: 2009-12-05.

[6] Rekurencja. Online, 2009. http://pl.wikipedia.org/wiki/Rekurencja, dostęp: 2010-02-22.

[7] Stanisław Ulam. Online, 2009. http://pl.wikipedia.org/wiki/Stanisław_Ulam, dostęp: 2009-10-21.

[8] Computer Music. Online, 2010. http://en.wikipedia.org/wiki/Computer_music, dostęp: 2010-02-02.

[9] Pierre Barbaud. Online, 2010. http://fr.wikipedia.org/wiki/Pierre_Barbaud, dostęp: 2010-01-24.

[10] Theodor W. Adorno. *Filozofia nowej muzyki*. Warszawa, 1974.

[11] Carlos Agon, Gérard Assayag, Jaccobo Baboni, Jean Bresson, Karim Haddad, Mathew Lima, Mikhail Malt. *OpenMusic User's Manual*. Paris, wydanie ircam documentation, 2004.

[12] Carlos Agon, Gérard Assayag, Jaccobo Baboni, Karim Haddad, Mathew Lima, Mikhail Malt. *OpenMusic Reference Manual*. Paris, wydanie ircam documentation, 2003.

[13] Carlos Agon, Gérard Assayag, Jean Bresson, redaktorzy. *The OM Composer's Book*, wolumen 1. Paris, 2006.

[14] Carlos Agon, Gérard Assayag, Olivier Delerue, Camilo Rueda. Objects, Time and Constraints in OpenMusic. Online. http://recherche.ircam.fr/equipe/repmus/RMPapers/CMJ98a/OMIC-MC98.html, dostęp: 2008-03-29.

[15] Carlos Agon, Gérard Assayag, Mikael Laurson, Camilo Rueda. Computer Assisted Composition at Ircam: PatchWork&OpenMusic. Online. http://recherche.ircam.fr/equipe/repmus/RMPapers/CMJ98/, dostęp: 2008-03-29.

[16] Torsten Anders. Composing Music by Composing Rules: Computer Aided Composition employing Constraint Logic Programming. Online, 2003. www.torsten-anders.de, dostęp: 2008-11-15.

[17] Torsten Anders. *Composing Music by Composing Rules: Design and Usage of a Generic Music Constraint System*. Praca doktorska, School of Music & Sonic Arts, Queen's University Belfast, 2007.

[18] Julian Anderson. In Harmony. Julian Anderson Introduces the Music and Ideas of Tristan Murail. *The Musical Times*, 134(1804):321–323, June 1993.

[19] Julian Anderson. A provisional History of Spectral Music. *Contemporary Music Review*, 19(2):7–22, 2000.

[20] Christopher Ariza. Navigating the landscape of computer aided algorithmic composition systems: a definition, seven descriptors, and a lexicon of systems and research. Online, 2005. www.flexatone.net/docs/nlcaacs.pdf, dostęp: 2009-10-05.

[21] Gérard Assayag. Computer Assisted Composition today. Online. http://recherche.ircam.fr/equipe/repmus/RMPapers/Corfou98/, dostęp: 2008-03-29.

[22] Gérard Assayag, Shlomo Dubnov, Olivier Delerue. Guessing the Composer's Mind: Applying Universal Prediction to Musical Style. Online. http://recherche.ircam.fr/equipe/repmus/RMPapers/ICMC99/, dostęp: 2008-03-29.

[23] James Bohn. Lejaren Hiller: A Total Matrix of Possibilities. Online, 2008. www.newworldrecords.org/uploads/filecLEfu.pdf, dostęp: 2009-11-02.

[24] Richard Boulanger, redaktor. *The Csound Book: perspectives in software synthesis, sound design, signal processing, and programming*. The MIT Press, Cambridge, Massauchusetts, 2000.

[25] Pierre Boulez. Technology and the Composer. *Orientations, Collected Writings*, strony 486–495. Cambridge, 1986.

[26] Jean Bresson, Carlos Agon, Gérard Assayag. OpenMusic5: A Cross-Platform Release of the Computer-Assisted Composition Environment. Online. www.articles.ircam.fr/textes/Bresson05b/, dostęp: 2008-03-29.

[27] Jean Bresson, Marco Stroppa, Carlos Agon. Symbolic Control of Sound Synthesis in Computer Assisted Composition. Online. http://recherche.ircam.fr/equipe/repmus/RMPapers/CMC99a/, dostęp: 2008-03-29.

[28] Jean Bresson, Marco Stroppa, Carlos Agon. Generetion and Representation of Data and Events for the Control of Sound Synthesis. *Proceedings Sound and Music Computing Conference*, Greece, 2007.

[29] Jim Bumgardner. Kircher's Mehcanical Composer: A Software Implementation. Online. http://krazydad.com/pubs/, dostęp: 2009-10-14.

[30] Norman Carey, David Clampitt. Self-Similar pitch structures, their duals, and rhytmic analogues. *Perspectives of New Music*, 34(2):63–87, Summer 1996.

[31] Grégoire Carpentier. *Approche computationnelle de l'orchestration musical*. Praca doktorska, Universite Paris VI - Pierre et Marie Curie, Grudzień 2008.

[32] Grégoire Carpentier, Damien Tardieu, Gérard Assayag, Xavier Rodet, Emmanuel Saint-James. An Evolutionary Approach to Computer-Aided Orchestration. Online. articles.ircam.fr/textes/Carpentier07a, dostęp: 2009-02-03.

[33] Pierre-Albert Castanet. Gérard Grisey and the Foliation of Time. *Contemporary Music Review*, 19(2):29–40, 2000.

[34] John Chowning, Max Mathews, Curtis Roads. Music Meets the Computer. Online, 2004. http://vodpod.com/watch/246252-music-meets-the-computer, dostęp: 05-02-2010.

[35] Anthony Cornicello. *Timbral Organization in Tristan Murail's Désintégration*. Praca doktorska, The Faculty of the Graduate School of Arts and Sciences Brandeis University, 2000. www.anthonycornicello.com, dostęp: 2004-04-21.

[36] Marc-André Dalbavie. Color. Partytura, 2003.

[37] Peter Desain. Lisp as a Second Language: Functional Aspects. *Perspectives of New Music*, 28(1):192–222, Winter 1990.

[38] Edsger W. Dijkstra. *Umiejętność programowania*. Warszawa, 1978.

[39] Chris Dobrian. Music Programming. An Introductory Essay. Online, 1988.

http://jmusic.ci.qut.edu.au/jmtutorial/CD.MusicProgramming.htm, dostęp: 2009-07-10.

[40] Mieczysław Drobner. *Akustyka muzyczna*. Kraków, 1973.

[41] Eric Drott. Timbre and the Cultural Politics of French Spectralism. *Proceedings of the Conference on Interdisciplinary Musicology*, Montreal, 2005. Online. www.oicm.umontreal.ca/doc/cim05/articles/DROTT_E_CIM05.pdf, dostęp: 2009-11-02.

[42] Umberto Eco. *Dzieło otwarte. Forma i nieokreśloność w poetykach współczesnych*. Warszawa, 1994.

[43] Joshua Fineberg. Guide to the Basic Concept and Techniques of Spectral Music-Appendix I. *Contemporary Music Review*, 19(2):81–113, 2000.

[44] Joshua Fineberg. Musical Examples - Appendix II. *Contemporary Music Review*, 19(2):115–134, 2000.

[45] Joshua Fineberg. Spectral Music. *Contemporary Music Review*, 19(2):1–5, 2000. Part 2.

[46] Cuthbert Girdlestone. *Jean-Philippe Rameau. His Life and Work*. New York, 1969.

[47] Richard Gregory. *Mózg i maszyny*. Warszawa, 2000.

[48] Gérard Grisey. Periodes. partytura, 1974.

[49] Gérard Grisey. Tempus ex Machina: A composer's reflection on musical time. *Contemporary Music Review*, 2:239–275, 1987.

[50] Gérard Grisey. Powiedziałeś: spektralny? *Glissando*, (1):2–3, wrzesień 2004. tłum.: Michał Mendyk, Marek Mroziewicz.

[51] Hanna Järveläinen. Algorithmic musical composition. Online, 2000. http://openpdf.com/ebook/musical-composition-pdf.html, dostęp: 2009-10-10.

[52] Chris Paul Harman. Beyond Computer-assisted-Composition: Son of the Sorcerer's Apprentice. *Ontario Notations*, strony 1–8, Summer 2004.

[53] Jonathan Harvey. Spectralism. *Contemporary Music Review*, 19(2):11–14, 2000.

[54] Marek Hołyński. *Sztuka i komputery*. Warszawa, 1976.

[55] Roman Ingarden. *Studia z estetyki t.II*, rozdział Utwór muzyczny i sprawa jego tożsamości. Warszawa, 1958.

[56] Hans Kaufmann. *Dzieje komputerów*. Warszawa, 1980.

[57] Michael Klingbeil. Software for Spectral Analysis, Editing, and Synthesis. Online, 2005. www.klingbeil.com/papers/spearfinal05.pdf, dostęp: 2008-08-02.

[58] Romana Kolarzowa. *Postmodernizm w muzyce*. Warszawa, 1993.

[59] Włodzimierz Kotoński. *Leksykon współczesnej perkusji*. Kraków, 1999.

[60] Włodzimierz Kotoński. *Muzyka elektroniczna*. Kraków, 2002.

[61] Tomasz Kręglewski, Piotr Misiurewicz, Krzysztof Sacha. *Przewodnik po technice mikrokomputerowej*. Warszawa, 1988.

[62] Marek Kłosiński. *Człowiek w sytuacji kontaktu z muzyką*. Warszawa, 1995.

[63] Mikael Laurson. Recent Development in PatchWork: PWConstraints - a Rule Based Approach to Complex Musical Problems. Online. www2.siba.fi/soundingscore/PDF/PWConstraints.pdf, dostęp: 2008-08-28.

[64] Mikael Laurson, Mika Kuuskankare. Extensible Constraint Syntax Through Score Accessor. Online. jim.afim-asso.org/jim2005/download/5.Extensible.pdf, dostęp: 2010-03-19.

[65] Grégoire Lorieux. Une analyse d'Amers de Kaija Saariaho. Online. www.univ-lille3.fr/revues/demeter/analyse/lorieux.pdf, dostęp: 2007-04-24.

[66] Witold Lutosławski. *Postscriptum*. Warszawa, 1999.

[67] Marija Masnikosa. *Muzički minimalizam*. Beograd, 2000.

[68] Joanna Miklaszewska. *Minimalizm w muzyce polskiej*. Kraków, 2003.

[69] Eduardo Reck Miranda. *Composing Music with Computers*. Focal Press, 2001.

[70] Małgorzata Moczurad, Włodzimierz Moczurad. Paradygmaty programowania. Online, 2006. http://wazniak.mimuw.edu.pl/index.php?title=Paradygmaty_programowania, dostęp: 2010-02-10.

[71] Tristan Murail. C'est un jardin secret. Partytura, 1977.

[72] Tristan Murail. After-thoughts. *Contemporary Music Review*, 19(2):5–9, 2000.

[73] Tristan Murail. Désintégration. Partytura, 2004.

[74] Tristan Murail. Scelsi and L'Itinéraire: The Exploration of Sound. *Contemporary Music Review*, 24(2/3):181–185, April/June 2005.

[75] Tristan Murail. Spectra and Sprites. *Contemporary Music Review*, 24(2/3):137–147, April/June 2005.

[76] Tristan Murail. Target Practice. *Contemporary Music Review*, 24(2/3):149–171, April/June 2005.

[77] Tristan Murail. The Revolution of Complex Sounds. *Contemporary Music Review*, 24(2/3):121–135, April/June 2005.

[78] Jerzy Nadolny. Harmonja duszy kosmicznej w świetle pythagorejsko-platońskich rozważań. *Kwartalnik Klasyczny*, VII(3), 1934.

[79] Gerhard Nierhaus. *Algorithmic Composition. Paradigms of Automated Music Generation*. Wien, 2009.

[80] Gilbert Nouno, Arshia Cont, Grégoire Carpentier, Jonathan Harvey. Making an orchestra speak. *Proceedings of the SMC 2009 - 6th Sound and Music Computing Conference*, strony 277–282, Portugal, July 2009.

[81] Jelena Novak. *Opera u doba media*. Novi Sad, 2007.

[82] Edward Ozimek. *Dźwięk i jego percepcja. Aspekty fizyczne i psychoakustyczne*. Warszawa-Poznań, 2002.

[83] Damien Pousset. The Works of Kaija Saariaho, Philippe Hurel and Marc-André Dalbavie-Stile Concertato, Stile Concitato, Stile Rappresentativo. *Contemporary Music Review*, 19:67–100, 2000. Part 2.

[84] Daniel Pressnitzer, Stephen McAdams. Acoustics, Psychoacoustics and Spectral Music. *Contemporary Music Review*, 19:33–59, 2000. Part 2.

[85] Charles Bodman Rae. *Muzyka Lutosławskiego*. Warszawa, 1996.

[86] François Rose. Introduction to the pitch organization of French spectral music. *Perspective of New Music*, (34/2):6–39, Summer 1996.

[87] Camilo Rueda, Magnus Lindberg, Mikael Laurson, Georges Bloch, Gérard Assayag. Integrating Constraint Programming in Visual Musical Composition Languages, June 1998. www.ircam.fr/equipes/repmus/RMPapers/CMJ98/, dostęp: 2009-07-10.

[88] Kaija Saariaho. Spins and Spells. Partytura, 1997.

[89] Juergen Schmitt. Gestalt revolution-a new approach to a unified view on structuring diverse levels of musical composition. Online. www.juergen-schmitt-komponist.de, dostęp: 2009-02-03.

[90] Diemo Schwartz, Dominique Virolle. SDIF. Format Specification. Online, 2006. recherche.ircam.fr/equipes/analyse-synthese/sdif/standard/sdif-standard.pdf, dostęp: 2009-07-11.

[91] Diemo Schwartz, Matthew Wright. Extensions and Applications of the SDIF Sound Description Interchange Format. Online. http://recherche.ircam.fr/equipes/analyse-synthese/schwarz/publications/icmc2000/, dostęp: 2009-07-11.

[92] K. Robert Schwarz. Steve Reich: Music as a Gradual Process: Part I-II. *Perspectives of New Music*, 19(1/2):373–392, 225–286, Autumn-Summer 1980-1981.

[93] Peter Seibel. Practical Common Lisp. Online. http://gigamonkeys.com/book/, dostęp: 2008-09-24.

[94] Barry Smart. *Postmodernizm*. Poznań, 1998.

[95] Ronald Bruce Smith. An Interview with Tristan Murail. *Computer Music Journal*, 24(1):11–19, Spring 2000.

[96] Guy Steele. Common Lisp the Language. Online. http://www.clm.pdf, dostęp: 10-03-2009.

[97] Stróżewski, Władysław. *Dialektyka twórczości*. Kraków, 1983.

[98] Martin Supper. A Few Remarks on Algorithmic Composition. *Computer Music Journal*, 25(1):48–53, Spring 2001.

[99] Krzysztof Szlifirski. *Angielsko-polski słownik terminologii nagrań dźwiękowych*. Warszawa, 2008.

[100] Mieczysław Szymczak, redaktor. *Słownik języka polskiego*. Warszawa, 1995.

[101] Władysław Tatarkiewicz. *Dzieje sześciu pojęć*. Warszawa, 1988.

[102] Heinrich Taube. An Introduction to Common Music. *Computer Music Journal*, 21(1):29–34, Spring 1997.

[103] Charlotte Truchet, Carlos Agon, Philippe Codognet. A Constraint Programming System for Music Composition, Preliminary Results. Online, 2001. http://www.ircam.fr/equipes/repmus/cpws/truchet.ps, dostęp: 2008-04-20.

[104] Charlotte Truchet, Gérard Assayag, Philippe Codognet. Visual and Adaptive Constraint Programming in Music. Online, 2001. http://www.ircam.fr/equipes/repmus/RMPPapers/TruchetICMC01.ps, dostęp: 2010-02-22.

[105] Hugues Vinet. Recent Research and Development at IRCAM. Online. articles.ircam.fr/textes/Vinet99d/, dostęp: 2009-02-03.

[106] Todd Winkler. *Composing Interactive Music. Techniques and Ideas Using Max*. The MIT Press, Cambridge, Massauchusetts, 2001.

[107] Piotr Wróblewski. *Algorytmy, struktury danych i techniki programowania*. Gliwice, 1997. wydanie drugie poprawione i uzupełnione.

[108] św. Augustyn. *Dialogi filozoficzne*, rozdział O muzyce. Kraków, 1999.