

## **Streszczenie:**

Poniższy dokument opisuje powstały projekt dyplomowy magisterski, którego celem było stworzenie aplikacji komponującej muzykę w dopasowaniu do zadanej linii melodycznej. W tym celu powstał *MusicAnalyzer*, program układający zapis nutowy akordów dla fortepianu, który może służyć jako akompaniament przy wykonywaniu utworu podanego na wejściu. Dokumentacja wykonanego projektu opisuje interfejs aplikacji, logikę działania programu, wykonane testy uzyskanego rozwiązania oraz ocenę jakości wskazaną przez osoby testujące.

## **Abstract:**

This document describes a graduation project, which aim was to create an application capable of composing an accompaniment to a melody read from input file. Thus *MusicAnalyzer* arose, a program creating music notation of chords for a piano, which can be played as an accompaniment to a melody from input file. The documentation of a given project describes user interface of a desktop application, algorithm composing the music, tests of few final result music compositions and their feedback given by people assessing program's efficiency.



## Spis treści

Streszczenie: .....	5
Abstract: .....	5
Spis treści.....	7
1. Wykaz stosowanych pojęć i skrótów .....	9
2. Wstęp i cel pracy .....	11
2.1. Wstęp .....	11
2.2. Cel pracy .....	11
3. Opis dziedziny .....	13
3.1. Opis tematu projektu dyplomowego.....	13
3.2. Podstawowe pojęcia muzyczne i funkcje harmoniczne .....	13
3.3. Przegląd dostępnych rozwiązań .....	22
3.3.1. Emily Howell .....	23
3.3.2. Mezzo .....	25
3.3.3. AthTek DigiBand .....	27
4. Opis rozwiązania .....	28
4.1. Opis interfejsu graficznego programu MusicAnalyzer.....	28
4.2. Informacje bazowe – pliki konfiguracyjne.....	33
4.3. Wczytywanie melodii wejściowej i metody określające jej cechy.....	34
4.3.1. Zdekodowanie sygnałów dźwięków .....	34
4.3.2. Wyszukanie zmian tonacji utworu .....	34
4.3.3. Wyszukanie zmian metrum .....	35
4.3.4. Ustalenie prawidłowej tonacji utworu .....	35
4.3.5. Ustalenie prawidłowych oznaczeń dźwięków.....	37
4.3.6. Ustalenie wartości rytmicznych wszystkich nut.....	37
4.4. Prezentacja graficzna zapisu nutowego.....	39
4.5. Algorytm tworzenia akompaniamentu .....	42
4.5.1. Harmony Search .....	42
4.5.2. Wyszukanie współbrzmień wejściowych i ustalenie ich atrybutów .....	45
4.5.3. Wybór węzłów dla akordów.....	48
4.5.4. Algorytm HS – tworzenie zbioru rozwiązań wejściowych .....	48
4.5.5. Algorytm HS – modyfikacje rozwiązań podczas iteracji.....	51

4.5.6.	Algorytm HS – funkcja oceny i warunki końcowe .....	53
4.5.7.	Ustalenie atrybutów nut akompaniamentu .....	55
4.6.	Zapis nutowy i dźwiękowy powstałego akompaniamentu .....	56
4.7.	Stosowane rozwiązania pomocnicze .....	58
4.7.1.	Sanford.Multimedia.....	58
4.7.2.	PSAMControlLibrary .....	59
4.7.3.	PSAMWPFCControlLibrary .....	59
5.	Ocena .....	61
5.1.	Założenia testów.....	61
5.2.	Kryteria oceny.....	62
5.3.	Wyniki testów.....	63
5.3.1.	I wariant – testy jakościowe .....	63
5.3.2.	II wariant – testy jakości przy zachowaniu wydajności.....	72
5.4.	Ocena jakości działania programu MusicAnalyze .....	81
6.	Podsumowanie .....	83
7.	Wykaz literatury .....	84
8.	Wykaz rysunków .....	86
9.	Wykaz tabel .....	89

## 1. Wykaz stosowanych pojęć i skrótów

1. *Test Turinga* - sposób mający dowodzić posiadania przez maszynę umiejętności myślenia na wzór procesu myślowego człowieka. Zaproponowany w 1950 roku przez Alan Turinga polega na prowadzeniu rozmowy z człowiekiem i konkurującą maszyną przez osobę mającą tu rolę sędziego. Jeśli nie jest on w stanie prawidłowo wskazać, które z nich jest maszyną, to przyjmuje się, że wówczas zdaje ona test.
2. *Progresja harmoniczna* – szereg współbrzmień następujących po sobie, które tworzą dobrze zharmonizowany układ i wrażenie spójnego motywu muzycznego u odbiorcy.
3. *MIDI* (ang. *Musical Instrument Digital Interface*) – kompletny system służący do komunikacji i wymiany informacji dźwiękowych pomiędzy elektronicznymi instrumentami muzycznymi. Pliki zapisane w tym standardzie mogą być odtwarzane na urządzeniach dzięki stałym próbkom dźwięku różnych instrumentów w nich zapisanych, sam plik zaś nie zawiera nagrania dźwięku w żadnej formie. Posiada za to kolekcję sygnałów, które odzwierciedlają różne muzyczne działania, takie jak włączenie / wyłączenie dźwięku o zadanej wysokości, naciśnięcie pedału w pianinie czy zmiana tempa utworu. W znajdującej się na początku sekcji nagłówkowej zawiera się także zestaw atrybutów pliku potrzebnych do jego prawidłowego odczytania i odtworzenia.
4. *Tessitura* – w odniesieniu do głosów instrumentalnych lub wokalnych, przedział dźwięków najczęściej używany w danej partii.
5. *Bitmapa* – obraz w grafice rastrowej, w którym określona jest jednoznacznie zawartość każdego piksela; umożliwia to szybkie i nieskomplikowane obliczeniowo wyświetlanie obrazu, lecz nie pozwala na zwiększenie dokładności w wyniku przybliżenia.
6. *Strój (muzyczny)* – system uporządkowania dźwięków w ramach jednej oktawy. Obecnie powszechnie stosowany jest strój równomiernie temperowany, który dzieli oktawę na 12 równych części. Dodatkowo strój muzyczny wyznacza częstotliwość dźwięków wydawanych przez instrumenty muzyczne. W okresie baroku, dźwięk A4 miał częstotliwość 415Hz, następnie zastąpiono go strojem 432Hz, a od okresu międzywojnia XX w. najczęściej używany jest strój 440Hz (stosowany w urządzeniach elektronicznych zgodnych ze standardem *MIDI*).
7. *HS* – algorytm sztucznej inteligencji *Harmony Search*
8. *Problem komiwojażera* – problem znalezienia najkrótszej ścieżki w grafie, która zahacza o wszystkie jego wierzchołki i wraca do punktu wyjścia. To zagadnienie należy do klasy problemów niedeterministycznie wielomianowych (ang. *NP*), gdyż nie istnieje żadna taka funkcja, która w sposób jednoznaczny jest w stanie wyznaczyć optymalne rozwiązanie dla dowolnego zestawu zmiennych wejściowych.
9. *Algorytm genetyczny* – algorytm optymalizacyjny z dziedziny sztucznej inteligencji, który wyszukuje rozwiązania w sposób naśladujący mechanizm ewolucji genetycznej organizmów żywych. Swoje działanie opiera na bazie osobników, które w wyniku krzyżowania, mutacji a następnie selekcji naturalnej ewoluują, powodując, że kolejne pokolenia są lepiej

dostosowane do pożądaných warunków. Jego użycie nie gwarantuje uzyskania najlepszego możliwego rozwiązania, a proces obliczeniowy może być długi i potrzebować wielu zasobów (baza osobników). Jednak znajduje on swoje zastosowanie w problemach, w których można łatwo ocenić jakość rozwiązania, lecz nie jest znany sposób osiągnięcia optymalnego wyniku. Przykładem takiego problemu jest wyszukiwanie minimum globalnego matematycznej funkcji wielu zmiennych.

## **2. Wstęp i cel pracy**

### **2.1. Wstęp**

Świat nauk ścisłych wielokrotnie w przeszłości stawiał sobie za cel zrozumienie i określenie stałymi regułami fenomenu osiągnięć nauk humanistycznych. Czy można w deterministyczny sposób opisać proces tworzenia przez poetę wiersza albo obrazu rysowanego przez malarza? Czy metody sztucznej inteligencji są w stanie naśladować, wydają się, cudownie twórczy proces zachodzący w mózgu artysty? Na te pytania naukowcy starają się odpowiadać twierdząco i popierają to coraz doskonalszymi osiągnięciami w tej dziedzinie, lecz na dzień dzisiejszy nie wydaje się, by los artystów został definitywnie przesądzony. Muzyka, podobnie jak inne dziedziny sztuki, dostarczają wrażeń (w tym przypadku dźwiękowych), które nie są jednoznacznie odczytywane przez wszystkich odbiorców. Dodatkowo, ocena tego samego utworu muzycznego w zależności od gatunku muzyki, w ramach którego go odbieramy, może się znacząco różnić.

Podobne wyzwania stają przed naukowcami pracującymi nad automatycznym komponowaniem muzyki. Problem ten można podzielić na dwa rodzaje wyzwań. Pierwsze podejście zakłada zupełne zastąpienie pracy kompozytora i umożliwienie tworzenia kompletnej muzyki od podstaw przez komputer. Drugim rodzajem zmagania z tworzeniem muzyki w sposób sztuczny jest układanie akompaniamentu do istniejącej już melodii, lecz w taki sposób, by obie ze sobą dobrze współbrzmiały. Problem ten wydaje się na pierwszy rzut oka nieco prostszy, gdyż posiadamy jakiś punkt zaczepienia, wzór, do którego staramy się dopasować. Jednakże osiągnięcie sukcesu w tym zadaniu niesie ze sobą podobne problemy, co tworzenie muzyki bez wzorca. Ponadto, to podejście wymaga szeregu założeń, które należy przyjąć, aby móc później ocenić jakość stworzonego dopasowania. Czy obie melodie mają być harmonicznym dopasowane? Jakiego typu harmonia ma je wiązać? Czy bardzo podobne do siebie melodie są pożądane, czy raczej oceniane jako nudne, nieciekawe? Czy dysonanse będą odbierane jako nieudolność programu czy raczej jako twórczy dodatek? Na te i wiele innych pytań, twórca oprogramowania tworzącego akompaniament musi sobie wcześniej odpowiedzieć a także zgodnie z nimi powinien być oceniany. Problemy z oceną jakości kompozycji przekładają się także na trudności z wytworzeniem ich, w szczególności gdy odpowiada za ten proces bezduszny program komputerowy.

### **2.2. Cel pracy**

Celem tego projektu było zmierzenie się z problemem automatycznego komponowania muzyki z wykorzystaniem technik sztucznej inteligencji bez posiadania szerokiej bazy wiedzy dziedzinowej. Jedyną muzyką znaną przez program komputerowy jest aktualnie wczytany utwór

z pliku wejściowego, do którego powstać ma akompaniament. Nie posiada on zbioru kompozycji, by móc się na nich wzorować, gdyż stawiane przed nim wyzwanie, to utworzenie pasującej melodii dzięki regułom harmonicznym i metodom sztucznej inteligencji. Tak powstał *MusicAnalyzer*, program tworzący własną muzykę przy wykorzystaniu algorytmu *Harmony Search* w oparciu o zasady klasycznej harmonii muzycznej. Utworzona przez niego nowa ścieżka dźwiękowa składa się z trzy- i czterodźwiękowych akordów, które tworząc podstawę harmoniczną mogą być rozszerzone o bardziej rozbudowaną melodię. Do działań w ramach pracy dyplomowej należy także dodać ocenę efektywności jego działań przy różnej parametryzacji algorytmu.



### 3. Opis dziedziny

#### 3.1. Opis tematu projektu dyplomowego

Tematem projektu dyplomowego jest automatyczne komponowanie muzyki w postaci tworzenia akompaniamentu do istniejącej melodii. Jak nietrudno się domyślić, jest to problem nieposiadający deterministycznego rozwiązania, które jesteśmy także w łatwy sposób ocenić, porównać z innym. Z tego względu, głównym obiektem pracy nad tym projektem była implementacja algorytmu z dziedziny sztucznej inteligencji (*Harmony Search*), który jest względnie nowym pomysłem i nie posiada wciąż wielu zastosowań. Wybrane podejście do automatycznego generowania muzyki wiąże się z kilkoma przyjętymi założeniami. W odróżnieniu od większości tego typu programów, *MusicAnalyzer* nie posiada bazy kompozycji stworzonych ludzką ręką, by na ich podstawie tworzyć własne utwory korzystając z pewnych modyfikacji i kombinacji. Jego główną siłą jest implementacja algorytmu harmonicznego, który ma za zadanie utworzyć nową ścieżkę dźwiękową będącą w harmonii z linią melodyczną wczytaną w pliku wejściowym. Szukanie prawidłowych współbrzmień a następnie ich ocena zostały oparte na zasadach klasycznej harmonii, którymi posługują się kompozytorzy od wieków. Efektem działań algorytmu jest sekwencja zmieniających się w kluczowych momentach utworu akordów. Element improwizacji i tworzenie wyszukanej wiodącej melodii celowo nie były zgłębiane, gdyż prace skupione były na szukaniu prawidłowego tła dla melodii już istniejącej. Program przystosowany jest do wczytania pliku z zapisem wielu ścieżek dźwiękowych jednocześnie i wszystkie z nich są jednakowo brane pod uwagę w szukaniu współbrzmienia z powstającym akompaniamentem.

#### 3.2. Podstawowe pojęcia muzyczne i funkcje harmoniczne

W opisywanym projekcie konieczne było zamodelowanie świata muzycznego, funkcji, figur oraz reguł współbrzmień, aby z nich zbudować własną kompozycję. Tworząc klasy odpowiadające pojęciom muzycznym, utworzono także reguły, które nimi rządzą w nomenklaturze stosowanej przez muzyków. Dzięki temu zabiegowi, proces kompozycji w programie *MusicAnalyzer* jest opisany pojęciami analogicznymi do tych używanych przez rzeczywistych kompozytorów. Na samym początku należy zatem przedstawić terminy ze świata teorii muzyki, które będą używane w dalszej części dokumentu.

- Tonalność - system uporządkowania dźwięków, dotyczący szczególnie harmoniki i melodyki. Tonalność charakteryzuje występowanie określonego centrum tonalnego<sup>1</sup> (głównego dźwięku) oraz zasad połączeń pomiędzy dźwiękami tworzącymi określone współbrzmienia.

---

<sup>1</sup> A. Poszowski, *Harmonia systemu tonalnego dur-moll*, Wydaw. Akademii Muzycznej w Gdańsku, Gdańsk 2001, s. 24

- Nuta – dźwięk w zapisie muzycznym; jego podstawowymi atrybutami jest wysokość (w pojęciu fizycznym odpowiadająca częstotliwości dźwięku) oraz długość trwania
- Półton – najmniejsza odległość między dźwiękami określona w teorii muzyki<sup>2</sup>. Dźwięk odległy o pół tonu od wskazanego jest większy bądź mniejszy o 1/12 częstotliwości dźwięku podstawowego (ta zależność nie dotyczy każdego stroju muzycznego, lecz ma zastosowanie w stroju równomiernie temperowanym, którym posługują się wszystkie urządzenia elektroniczne i jak strojone są obecnie instrumenty).
- Krzyżyk – oznaczenie podwyższenie nuty o pół tonu względem miejsca, w którym się znajduje na pięciolinii, krzyżyk dodany do nuty (za wyjątkiem dźwięków H i E) wskazuje dźwięki odpowiadające czarnym klawiszom na fortepianie.
- Bemol – ma działanie odwrotne niż krzyżyk, obniża wskazany dźwięk o pół tonu.
- Oktawa – odległość między dźwiękami odpowiadająca 12-stu półtonom, co daje dwukrotnie większą lub mniejszą częstotliwość pomiędzy przykładowymi dźwiękami<sup>3</sup>, należącymi zatem do tego samego szeregu harmonicznego. Ponieważ wszystkie dźwięki w ramach jednej oktawy mają swoich odpowiedników w pozostałych oktawach poprzez zwielokrotnienie bądź podzielenie ich częstotliwości, z punktu widzenia harmonii brzmień, są one tak samo ważne i można je stosować zamiennie w zależności od potrzeby, bądź instrumentu, który daną melodię wykonuje. W programie *MusicAnalyzer* kompozycja w pierwszej kolejności ustala konkretne dźwięki w ramach oktawy, a następnie dopiero dobiera oktawę, w ramach której mają być one dopasowane, aby możliwie najlepiej akompaniowały melodię podstawową.
- Oznaczenia nut w ramach oktawy – kolejne dźwięki w zapisie muzycznym posiadają swoje unikatowe oznaczenia. W notacji używanej w tym dokumencie, są one poukładane alfabetycznie za wyjątkiem dźwięku *H*, który występuje między *A* i *C* (w amerykańskim nazewnictwie można w jego miejscu spotkać *B*)<sup>4</sup>. Podstawowe dźwięki (odpowiadające białym klawiszom fortepianu) są oddalone względem siebie (różnica wysokości dźwięku) o odległość całego bądź połowy tonu. Układ ten zatem nie jest spójny, ale powtarzalny w przypadku niższych i wyższych oktaf. Zestaw dźwięków i wzajemnych odległości przedstawia poniższy rysunek.

---

<sup>2</sup> *Zanim zaczniesz grać na... : ABC muzyki*, Polskie Wydawnictwo Muzyczne, Kraków 2002, s.12

<sup>3</sup> Andrzej Rakowski, *Percepcja wysokości dźwięku*, Państwowa Wyższa Szkoła Muzyczna, Warszawa 1978, s. 47

<sup>4</sup> *Zanim zaczniesz grać na...*, dz. cyt., s.14

## GAMA C-DUR

POWSCIAGLIWY BLOG MUZYCZNY.BLOGSPOT.COM



Rysunek 3.1 Dźwięki gamy C-dur, źródło <http://3.bp.blogspot.com/-ZbOavfTrYJg/T1QALqWFwdI/AAAAAAAAADg/eC5xMCH9c6g/s1600/gama-c-dur-odleglosci.png>

Dodatkowo, aby oznaczyć dźwięki znajdujące się pomiędzy tymi odległymi o cały ton (odpowiadające czarnym klawiszom fortepianu), stosuje się krzyżyki i bemole<sup>5</sup>, gdyż na pięciolinii zapisane są na tej samej wysokości. I tak nazewnictwo prezentuje się następująco: dla przykładu, dźwięk C z krzyżykiem to Cis, natomiast D z bemolem to Des.

- Absolutne oznaczenie wysokości dźwięku – do opisanego powyżej schematu nazywania dźwięków w ramach danej oktawy, należy dodać jeszcze jej numer, do której z nich należy. Człowiek posiada ograniczoną umiejętność słyszenia dźwięków o bardzo niskich bądź wysokich częstotliwościach<sup>6</sup>. Stąd też zakres oznaczeń nie jest zbyt duży i dostosowany jest do możliwości ludzkiego ucha. Najniższa słyszalna oktawa posiada numer 0 (dla przykładu dźwięk C0 ma częstotliwość 16,35 Hz<sup>7</sup>), natomiast najwyższe częstotliwości słyszane przez człowieka to oktawa 10 (C10  $\approx$  16 700 Hz<sup>8</sup>).
- Pięciolinia – układ pięciu linii służący do określenia częstotliwości zapisanych na niej dźwięków (dźwięk pokazany wyżej ma wyższą częstotliwość, a niżej – mniejszą). Z tego powodu wywiązał się zwyczaj określania nut jako wysokie lub niskie. Mogą być zapisywane na liniach lub pomiędzy nimi i odpowiadają dźwiękom wydobywanym przez białe klawisze na fortepianie<sup>9</sup>.

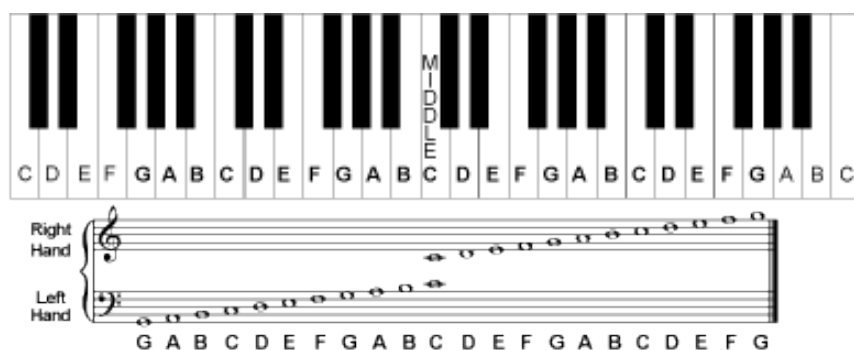
<sup>5</sup> Zanim zaczniesz grać na..., dz. cyt., s.38

<sup>6</sup> Andrzej Rakowski, *Percepcja wysokości...*, dz. cyt., s. 24

<sup>7</sup> <https://pl.wikipedia.org/wiki/MIDI>, (dostęp 14.09.2016)

<sup>8</sup> <https://pl.wikipedia.org/wiki/MIDI>, (dostęp 14.09.2016)

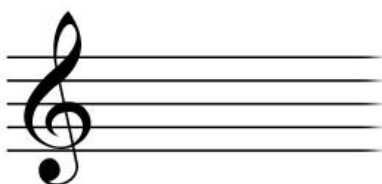
<sup>9</sup> Zanim zaczniesz grać na..., dz. cyt., s.7



Rysunek 3.2 Układ klawiszy fortepianu i odpowiadających im dźwięków na pięcioliniach, źródło: <https://tucsonsongstress.files.wordpress.com/2012/11/staffnkeys.gif>

Następstwo dźwięków po sobie obrazuje się poprzez umieszczenie ich obok siebie (od lewej do prawej). Tak też nuty, które mają zabrzmieć jednocześnie, zaznacza się jedna nad drugą.

- Klucz wiolinowy – znak stawiany na początku pięciolinii określający wysokość znajdujących się na niej dźwięków<sup>10</sup>. Klucz wiolinowy znajduje się na drugiej linii i wyznacza na niej dźwięk G4 (392 Hz<sup>11</sup>).



Rysunek 3.3 Klucz wiolinowy, źródło <http://www.music-paper.com/ImagesClefs/treble-clef.jpg>

- Klucz basowy – analogicznie jak wiolinowy, określa wysokość dźwięków na pięciolinii. Służy do zapisu niższych dźwięków, umieszczony na trzeciej linii wskazuje nutę F3 (ok. 175 Hz<sup>12</sup>)



Rysunek 3.4 Klucz basowy, źródło: [http://www.theyrenotourgoats.com/wp-content/uploads/2014/10/Bass\\_Clef.jpg](http://www.theyrenotourgoats.com/wp-content/uploads/2014/10/Bass_Clef.jpg)

- Akord – jednoczesne współbrzmienie kilku dźwięków (przynajmniej trzech) o różnej wysokości.

<sup>10</sup> Zanim zaczniesz grać na..., dz. cyt., s.18

<sup>11</sup> <https://pl.wikipedia.org/wiki/MIDI>, (dostęp 14.09.2016)

<sup>12</sup> <https://pl.wikipedia.org/wiki/MIDI>, (dostęp 14.09.2016)

- Konsonans – współbrzmienie przynajmniej dwóch dźwięków, które jest odbierane słuchowo jako zgodne. Ma ono swoje uzasadnienie czysto fizyczne, gdyż nasz słuch pozytywnie odbiera dźwięki, których częstotliwości nakładają się na siebie, mają względnie niewielką wspólną wielokrotność<sup>13</sup>. Gdy wybrzmiewają jednocześnie, fale akustyczne na siebie nachodzą, wzmacniają się i zachodzi zjawisko dudnienia.
- Dysonans – jest przeciwieństwem konsonansu, a zatem współbrzmieniem niedoskonałym, które sprawia wrażenie niezgodności. W muzyce współczesnej panuje coraz większe przyzwolenie na stosowanie takich współbrzmień, co jednak bywa kontrowersyjne i odrzucane przez część twórców oraz odbiorców<sup>14</sup>. Daje to jednak szersze pole działania kompozytorom, gdyż umożliwia tworzenie nieznanych dotąd melodii. W projekcie *MusicAnalyzer* preferowane są jednakowoż klasyczne współbrzmienia harmoniczne, które nie zawierają dysonansów.
- Interwał – określenie odległości między dwoma dźwiękami stosowane w muzykologii. Nazwy interwałów pochodzą z języka łacińskiego<sup>15</sup> i można je podzielić na proste<sup>16</sup> (ich rozpiętość mieści się w obrębie oktawy) i złożone (pozostałe, większe). Do interwałów prostych należą:
  - pryma – odległość zawarta między powtórzonymi stopniami skali
  - sekunda mała – 1 półton
  - sekunda wielka – 2 półtony
  - tercja mała – 3 półtony
  - tercja wielka – 4 półtony
  - kwarta – 5 półtonów
  - tryton – 6 półtonów
  - kwinta – 7 półtonów
  - seksta mała – 8 półtonów
  - seksta wielka – 9 półtonów
  - septyma mała – 10 półtonów
  - septyma wielka – 11 półtonów
  - oktawa – 12 półtonów
- Skala durowa - to siedmiostopniowa skala z charakterystycznym półtonem między stopniami III i IV oraz VII i VIII (stanowiącym powtórzenie I stopnia o oktawę wyżej)<sup>17</sup>. Wszystkie dźwięki zawierają się w ramach jednej oktawy. Uznaje się, że skala durowa i utwory na niej oparte mają radosne brzmienie.

<sup>13</sup> Andrzej Rakowski, *Percepcja wysokości...*, dz. cyt., s. 67

<sup>14</sup> Andrzej Rakowski, *Percepcja wysokości...*, dz. cyt., s. 72

<sup>15</sup> Irena Poniatowska, *Dzieło muzyczne, teoria, historia, interpretacja*, PWM, Kraków 1984, s. 74

<sup>16</sup> *Zanim zaczniesz grać na...*, dz. cyt., s.26

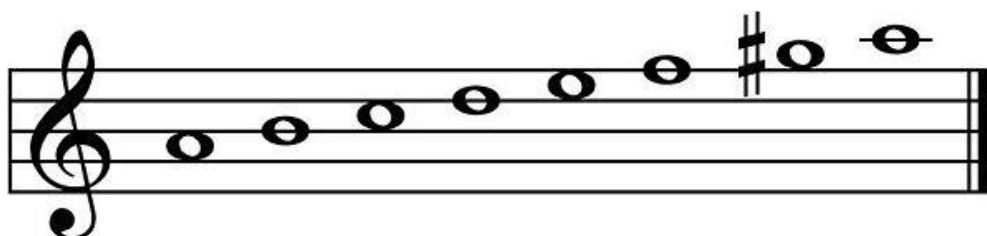
<sup>17</sup> A. Poszowski, *Harmonia systemu tonalnego...*, dz. cyt., s. 37

- Skala molowa eolska - to siedmiostopniowa skala z charakterystycznym półtonem między stopniami II i III<sup>18</sup>. Wszystkie dźwięki zawierają się w ramach jednej oktawy. Skale molowe i oparte na nich fragmenty utworów muzycznych mają smutne brzmienie.



Rysunek 3.5 Skala molowa eolska w gamie na przykładzie gamy a – moll, źródło: [http://3.bp.blogspot.com/-c1\\_q-799WgM/T1QOY7uNJGI/AAAAAAAACjI/hpUDBezDOs4/s400/gama-a-moll-odleglosci.png](http://3.bp.blogspot.com/-c1_q-799WgM/T1QOY7uNJGI/AAAAAAAACjI/hpUDBezDOs4/s400/gama-a-moll-odleglosci.png)

- Skala molowa harmoniczna – inna skala molowa, charakteryzująca się względem eolskiej tym, iż posiada podwyższony VII stopień o pół tonu. W efekcie półtony występują między II i III stopniem oraz VII i VIII, a między VI i VII występuje interwał sekundy zwiększonej (3 półtony)<sup>19</sup>. Ta skala jest najbardziej popularna wśród kompozycji molowych, gdyż podwyższony VII stopień przyjmuje funkcję dźwięku wyraźnie prowadzącego do rozwiązania, akordu toniki.



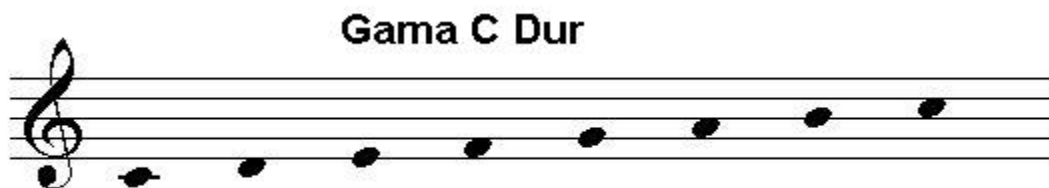
Rysunek 3.6 Skala harmoniczna na przykładzie gamy a - moll, źródło: [https://upload.wikimedia.org/wikipedia/commons/thumb/4/4b/Moll\\_harm.jpg/600px-Moll\\_harm.jpg](https://upload.wikimedia.org/wikipedia/commons/thumb/4/4b/Moll_harm.jpg/600px-Moll_harm.jpg)

- Gama - to szereg dźwięków ułożonych według reguł danej skali muzycznej, zaczynający się od konkretnego dźwięku, który zazwyczaj nadaje nazwę tak utworzonej gamie<sup>20</sup>. Zawiera ona każdorazowo siedem dźwięków, które mogą należeć do różnych oktaf (zaczynać się na różnych wysokościach), natomiast ich wzajemne odległości są stałe i charakterystyczne dla danej skali.

<sup>18</sup> A. Poszowski, *Harmonia systemu tonalnego...*, dz. cyt., s. 43

<sup>19</sup> A. Poszowski, *Harmonia systemu tonalnego...*, dz. cyt., s. 46

<sup>20</sup> *Zanim zaczniesz grać na...*, dz. cyt., s.34



*Rysunek 3.7 Dźwięki należące do gamy C - dur zapisane w kluczu wiolinowym*

- Funkcja harmoniczna - każda tonacja posiada charakterystyczne dla siebie współbrzmienia akordów, na których zbudowane są kompozycje w danej tonacji<sup>21</sup>. W tym rozumieniu, akordy posiadają swoją funkcję harmoniczną, gdyż w ramach danej tonacji budują lub rozładują napięcie, sprawiają wrażenie stabilności lub chaosu w danej frazie. Te same akordy występujące w różnym kontekście są odbierane w inny sposób, stąd też przyporządkowane im funkcje harmoniczne różnią się w zależności od tonacji, w jakiej występują.
- Trójdźwięk durowy – składa się z trzech dźwięków, które oddalone są od siebie w następujący sposób<sup>22</sup>:
  - 1. i 2. – o tercję wielką
  - 2. i 3. – o tercję małą
  - 3. i 1. – o kwartę czystą

Wiodąca tercja wielka powoduje, iż współbrzmienie odbierane jest jako pozytywne (durowe). Opisana powyżej kolejność interwałów odpowiada akordowi bez przewrotu.

- Trójdźwięk molowy – składa się z trzech dźwięków, które oddalone są od siebie w następujących interwałach<sup>23</sup>:
  - 1. i 2. – o tercję małą
  - 2. i 3. – o tercję wielką
  - 3. i 1. – o kwartę czystą

Tercja mała na początku akordu skutkuje brzmieniem odbieranym jako smutne (molowe). Przedstawiona powyżej kolejność odpowiada akordowi bez przewrotu.

- Przewrót akordu – ta cecha akordu oznacza kolejność dźwięków, z jakich się składa. Każdy akord posiadający swoją funkcję harmoniczną ma określoną wzajemną kolejność nut występujących po sobie<sup>24</sup>. Możliwa jest natomiast zmiana dźwięku, od którego akord ma się zaczynać. W efekcie uzyskujemy inne brzmienie, które jednak zachowuje swoje

<sup>21</sup> A. Poszowski, *Harmonia systemu tonalnego...*, dz. cyt., s. 78

<sup>22</sup> Irena Poniatowska, *Dzieło muzyczne...*, dz. cyt., s. 226

<sup>23</sup> Irena Poniatowska, *Dzieło muzyczne...*, dz. cyt., s. 227

<sup>24</sup> Piotr Kałużny, *Skale muzyczne we współczesnej harmonii tonalnej*, Wydawnictwo Akademii Muzycznej, Poznań 1994, s.15

znaczenie harmoniczne. Akord o swojej podstawowej budowie jest określany jako „bez przewrotu”. Możliwe przewroty akordu C-dur prezentuje poniższy rysunek.



Rysunek 3.8 Akord C-dur w przewrotach, źródło: <http://www.cyja.webd.pl/akordeon/images/stories/15-01%20przewroty%20c.jpg>

- Tonika – podstawowy trójdźwięk każdej gamy zbudowany z dźwięków na I, III i V stopniu gamy. Jest ośrodkiem tonalnym, który rozładowuje napięcie zbudowane wewnątrz frazy<sup>25</sup>, stąd najczęściej znajduje się na ich końcu lub też na początku. W gamie molowej jest molowy, natomiast w durowej – durowy.
- Subdominanta – akord pełniący tę funkcję harmoniczną zbudowany jest z dźwięków na IV, VI i VIII stopniu gamy. We frazach muzycznych jest wykorzystywany jako akord przejściowy. Ze względu na swoją bliskość do akordu toniki, wykorzystywany jest często w muzyce kościelnej<sup>26</sup>, gdyż w zestawieniu z toniką tworzy poważnej, dostojne wrażenie. W gamie molowej jest molowy, natomiast w durowej – durowy.
- Dominanta – akord o budowie tercjowej zbudowany na V stopniu gamy, uzupełniony o stopień VII i II. Jest bardzo charakterystyczną funkcją harmoniczną, występuje niejako w opozycji do ośrodka tonalnego (akordu toniki), ze względu na zawarte w niej stopnie bezpośrednio sąsiadujące z toniką (I stopniem gamy)<sup>27</sup>. Pojawienie się jej w danej frazie bywa często momentem kulminacyjnym i zwiastuje rozładowanie zbudowanego napięcia. W gamach molowych często stosuje się skalę harmoniczną, by móc wykorzystać wówczas akord dominanty, w którym podwyższony VII stopień gamy jest oddalony od toniki zaledwie o pół tonu.
- Dominanta septymowa – akord dominanty z dodanym IV stopniem gamy, który wraz z V stopniem gamy, na którym akord jest zbudowany, tworzy interwał septymy małej. Zastosowanie tego akordu potęguje wyrazistość dominanty<sup>28</sup> i powoduje, że jeszcze wyraźniej prowadzi ona do rozwiązania na tonikę.
- Triada harmoniczna (inaczej akordy główne) - trzy podstawowe trójdźwięki zbudowane na pierwszym (tonika), czwartym (subdominanta) i piątym stopniu gamy (dominanta).

<sup>25</sup> Jacek Targosz, *Podstawy harmonii funkcyjnej*, PWM, Kraków 2011, s. 16

<sup>26</sup> Jacek Targosz, *Podstawy harmonii...*, dz. cyt., s. 19

<sup>27</sup> Jacek Targosz, *Podstawy harmonii...*, dz. cyt., s. 21

<sup>28</sup> Jacek Targosz, *Podstawy harmonii...*, dz. cyt., s. 25





*Rysunek 3.9 Akordy triady harmoniczej w gamie C - dur*

- Znaki przy kluczu – są to krzyżyki bądź bemole zapisane przy kluczu na początku pięciolinii, które obowiązują dla wskazanych nutach do odwołania. Zaznaczane są na odpowiedniej wysokości i odnoszą się do wszystkich nut o danej wysokości dźwięku obniżając go lub podwyższając o pół tonu<sup>29</sup>. Dzięki temu oznaczeniu na początku systemu, nie jest konieczne prezentowanie znaków przy każdej nucie. Dodatkowo oznaczenie to informuje nas w jakiej tonacji zapisany jest utwór, ponieważ każdy zestaw znaków jest przyporządkowany tylko do jednej tonacji durowej i molowej. Podstawowa para gam (C-dur i a-moll) jako jedyne nie posiadają znaków przy kluczu.



*Rysunek 3.10 Znaki przy kluczu wiolinowym, źródło:*

*[https://upload.wikimedia.org/wikipedia/commons/thumb/8/84/A-major\\_f-sharp-minor.svg/150px-A-major\\_f-sharp-minor.svg.png](https://upload.wikimedia.org/wikipedia/commons/thumb/8/84/A-major_f-sharp-minor.svg/150px-A-major_f-sharp-minor.svg.png)*

- Metrum – schemat opisujący rytm w utworze, obręb każdego taktu. Określa układ akcentów oraz wartości rytmiczne, na które każdy takt się dzieli<sup>30</sup>. W połączeniu z określeniem tempa taktowania utworu, pozwala określić długość trwania każdej z nut. Metrum podaje się poprzez określenie ile oraz jakich wartości rytmicznych mieści się w jednym takcie.

<sup>29</sup> Irena Poniatowska, *Dzieło muzyczne...*, dz. cyt., s. 186

<sup>30</sup> Witold Rudziński, *Nauka o rytmie muzycznym. C 1*, PWM, Kraków 1987, s. 14



Rysunek 3.11 Przykłady metrum i ich wyjaśnienie, źródło: [http://e-perkusja.pl/plik/nauka\\_gry\\_na\\_perkusji/Artykuly/Czytanie\\_z\\_nut/04\\_Metrum\\_gorna\\_cyfra.jpg](http://e-perkusja.pl/plik/nauka_gry_na_perkusji/Artykuly/Czytanie_z_nut/04_Metrum_gorna_cyfra.jpg)

- Takt – określony poprzez metrum odcinek czasowy, najmniejsza jednostka podziału rytmicznego utworu<sup>31</sup>. Na pięciolinii znakiem końca taktu i zarazem początku kolejnego jest pojedyncza kreska taktowa.
- Wartość rytmiczna nuty – oznaczenie pomagające określić względną długość trwania nut w utworze. Ich bezwzględna długość trwania zależy oczywiście od tempa wykonywanego utworu i może być zmienna, lecz przyporządkowanie wartości rytmicznych pozwala zapisać stałą zależność długości trwania zachodzącą między poszczególnymi nutami<sup>32</sup>. Dostępne wartości odpowiadają zwielokrotnieniom mniejszych nut w ilości odpowiadającej potęgze liczby 2<sup>33</sup>. Do podstawowych oznaczeń rytmicznych należą:
  - cała nuta (równa dwóm półnutom)
  - półnuta (równa dwóm ćwierćnutom)
  - ćwierćnuta (równa dwóm ósemkom)
  - ósemka (równa dwóm szesnastkom)
  - szesnastka (równa dwóm trzydziestodwójkom)
  - trzydziestodwójka (równa dwóm sześćdziesięcioczwórkom)
  - itd...

### 3.3. Przegląd dostępnych rozwiązań

Automatyczne komponowanie muzyki jest wyzwaniem, z którym naukowcy działający w obszarze sztucznej inteligencji mierzą się od 20 - 30 lat. Powstało wiele rozwiązań, które starają się naśladować genialnego w swej pracy rzeczywistego kompozytora, który potrafi stworzyć dzieło przyjemne dla słuchacza, dodatkowo zawierające swój unikatowy styl<sup>34</sup>. Prace w tej

<sup>31</sup> Witold Rudziński, *Nauko o rytmie...*, dz. cyt., s. 10

<sup>32</sup> Irena Poniątkowska, *Dzieło muzyczne...*, dz. cyt., s. 124

<sup>33</sup> Andrzej Rakowski, *Percepcja wysokości...*, dz. cyt., s. 27

<sup>34</sup> G. Johnson, *Undiscovered Bach? No, a Computer Wrote It*, 11.11.1997, <http://www.nytimes.com/1997/11/11/science/undiscovered-bach-no-a-computer-wrote-it.html?pagewanted=all>, dostęp 20.08.2016 r.

dziedzinie są na tyle zaawansowane, że powstały już rozwiązania komercyjne umożliwiające generowanie muzyki z bardzo dobrymi efektami, przydatnymi dla osób próbujących układać własne kompozycje muzyczne. Niektóre rozwiązania użyte w programie *MusicAnalyzer* mają swoje odpowiedniki w przedstawionych poniżej przykładach.

### **3.3.1.Emily Howell**

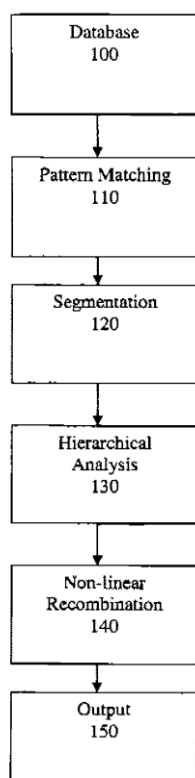
*Emily Howell*<sup>35</sup> to nazwa programu komputerowego autorstwa Davida Cope'a powstałego w latach 90-tych na uniwersytecie w Santa Cruz w Kalifornii. Projekt jest kontynuacją pracy uczonego nad programem *EMI* (*Experiments in Musical Intelligence*). Zasilany był bazą utworów wybranego kompozytora a następnie tworzył nową kompozycję, która stylistyką nawiązywała do utworów z zestawu wejściowego. *Emily Howell* funkcjonuje w analogiczny sposób, natomiast jako dane wejściowe posiada jedynie utwory wytworzone przez *EMI*, dzięki czemu, przynajmniej według jego twórcy<sup>36</sup>, wykształciła własny styl kompozytorski.

Szczegóły algorytmu zastosowanego w programie *EMI* nie są znane, natomiast wiadomo, iż jest on oparty na kilku krokach:

---

<sup>35</sup> <http://artsites.ucsc.edu/faculty/cope/Emily-howell.htm>, dostęp 20.08.2016 r.

<sup>36</sup> Jacqui Cheng, *Virtual composer makes beautiful music - and stirs controversy*, 30.09.2009, <http://arstechnica.com/science/2009/09/virtual-composer-makes-beautiful-musicand-stirs-controversy/>, dostęp 20.08.2016 r.



- 1) Utworzenie bazy źródłowej kompozycji, których styl chcemy naśladować
- 2) Znalezienie wzorców (motywów dźwiękowych i rytmicznych)
- 3) Podział dzieła na segmenty zgodnie ze znalezionym rytmem
- 4) Analiza hierarchiczna – wszystkim współbrzmiom przypisywane są ich funkcje, jakie pełnią w danej frazie; pomaga to określić np. które akordy po sobie występują;
- 5) Nieliniowe rekombinacje i modyfikacje powstających fragmentów z posiadanych wzorców
- 6) Wybór najlepszego utworu i zwrócenie go na wyjście programu

FIG. 1

Rysunek 3.12 Schmeta działania Emily Howell, źródło <https://www.google.com/patents/US7696426>

Dodatkowo program jest wciąż ulepszany, gdyż powstające utwory są oceniane przez dr Cope'a i wyniki jego szczegółowej oceny wpływają na proces tworzenia kolejnych wyników.

Rozwiązania stosowane w obu projektach okazały się na tyle wydajne, iż David Cope posiadał patent<sup>37</sup> na terenie Stanów Zjednoczonych chroniący jego autorski algorytm komponowania muzyki. Ponadto program *EMI* w 1997 roku na Uniwersytecie w Oregonie zmierzył się w teście<sup>38</sup> przypominającym muzyczny odpowiednik Testu Turinga. Na podstawie posiadanej bazy utworów J. S. Bacha stworzył własną kompozycję (miniaturę fortepianową) imitującą styl geniusza epoki baroku. Podobne zadanie otrzymał szanowany, współczesny kompozytor Steve Larson. Następnie pianistka wykonała przed oceniającą publicznością powyższe utwory oraz oryginalny utwór Bacha. Zebrani mieli za zadanie ocenić, która z kompozycji jest oryginałem, która stworzona przez współczesnego kompozytora, a która przez program *EMI*. Ich werdykt był bardzo zaskakujący, gdyż to komputerowo wytworzony utwór został uznany za oryginalne dzieło z XVII wieku, za to utwór Larsona wzięto za wytwór sztucznej

<sup>37</sup> <https://www.google.com/patents/US7696426>, dostęp 22.08.2016 r.

<sup>38</sup> G. Johnson, *Undiscovered Bach...*, dz. cyt.

inteligencji, natomiast pierwotna kompozycja Bacha została odebrana jako imitacja autorstwa współczesnego kompozytora.

Wyniki przedstawionego powyżej konkursu nie dają nam pewności co do pozytywnej oceny walorów artystycznych kompozycji programu *EMI*, gdyż publiczność w swoim głosowaniu źle oceniła utwór samego Bacha. Można jednak sądzić, że efekty obliczeń *Emily Howell* można przyjąć za dobrej jakości muzykę, ponieważ doczekały się one dwóch albumów na płytach: *From Darkness, Light* (2009) i *Breathless* (2012). Niemniej jednak, sztuczne tworzenie muzyki jest wciąż źle odbierane przez środowisko artystyczne, przez co wielu muzyków odmawia wykonywania utworów autorstwa *Emily*.

### 3.3.2. Mezzo

W 2012 roku na tej samej uczelni, na której działał David Cope, pojawił się kolejny projekt z gatunku automatycznego komponowania muzyki. Wówczas Daniel Lankford Brown ukończył swój przewód doktorski o temacie *Expressing narrative function in adaptive, computer-composed music*<sup>39</sup>. W ramach swoich badań stworzył program komputerowy *Mezzo*, który generuje na żywo tło muzyczne do dowolnej gry komputerowej prowadzonej przez użytkownika komputera. Powstająca muzyka pochodzi z gatunku neoromantyzmu, gdyż zgodnie z zamierzeniem, zasady harmonii i komponowania melodii zapisane w programie były konsultowane z takimi muzykologami jak V. Kofi Agawu, M. Grabócz, W. Caplin, R. Hatten czy B. Almén.

Projekt ten starał się sprostać przede wszystkim dwóm wyzwaniom. Jak stworzyć melodię, która ciągle ewoluuje, lecz mimo powtarzanych czynności przez gracza nie staje się nudna? Drugim wyzwaniem jest problem dopasowania muzyki do emocji przeżywanych przez gracza. Według autora, można im sprostać, gdy sposób powtarzania motywów muzycznych będzie czymś więcej niż repetycją, gdy zawarte w nich frazy i dźwięki będą miały określone funkcje, które będą się stopniowo zmieniać, ewoluować a niektóre z nich będą się wyróżniać tworząc główną linię melodyczną.

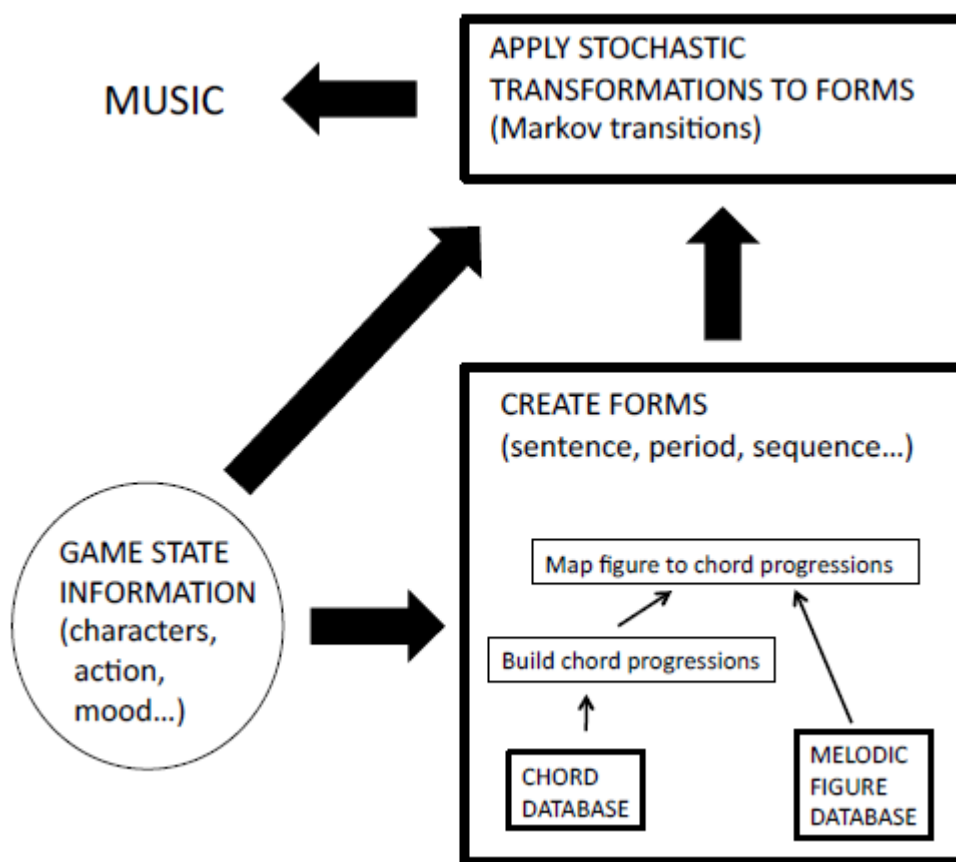
Tworzenie własnych kompozycji opiera się wprawdzie na wczyciu istniejących motywów muzycznych oraz progresji harmonicznym. Następnie przy pomocy algorytmu genetycznego powstają indywidualne progresje tworzone jednak na wzór tych wejściowych, które dzięki temu posiadają wspólny styl muzyczny. Wówczas, kiedy gracz uruchamia grę i w nią gra, program w czasie rzeczywistym stara się dopasować posiadane frazy muzyczne do odpowiednich momentów w grze, zachowując przy tym ich ciągłość, łagodne przejścia i podobieństwo między kolejnymi motywami. W ramach kolejnych fraz generowane są kolejne zaburzenia, kulminacje, aby zsynchronizować je z tym, co się dzieje w grze. Dla przykładu, w odpowiedzi na osiągnięcie sukcesu przez gracza, pojawia się progresja wstępująca, która odzwierciedla emocje związane radością i tryumfem. Analogicznie, w momencie porażki pojawia się melodia zstępująca, często

---

<sup>39</sup> D. L. Brown, *Expressing narrative function in adaptive, computer-composed music*, 06.2012, [http://www.danielbrownmusic.com/uploads/1/3/2/3/13234393/final\\_dissertation\\_final\\_edit.pdf](http://www.danielbrownmusic.com/uploads/1/3/2/3/13234393/final_dissertation_final_edit.pdf), dostęp 22.08.2016 r.

w tonacji molowej, która daje odczucie zawiedzenia, straconej szansy. Aby uniknąć zbyt częstych powtarzalności tych modyfikacji, są one dobierane stochastycznie: możliwe transformacje są zamodelowane w postaci łańcuchów Markowa. Powtórzenia motywów i przejścia między określonymi frazami są oznaczone pewnym prawdopodobieństwem (zgodnie z parametrami dla danego fragmentu gry), przez co modyfikacja regresji harmonicznym nawet w tym samym fragmencie gry jest za każdym razem nieco inna.

Poniższy schemat przedstawia sposób generowania akompaniamentu muzycznego przez program *Mezzo* podczas gry.



Rysunek 3.13 Schemat działania programu *Mezzo*, źródło

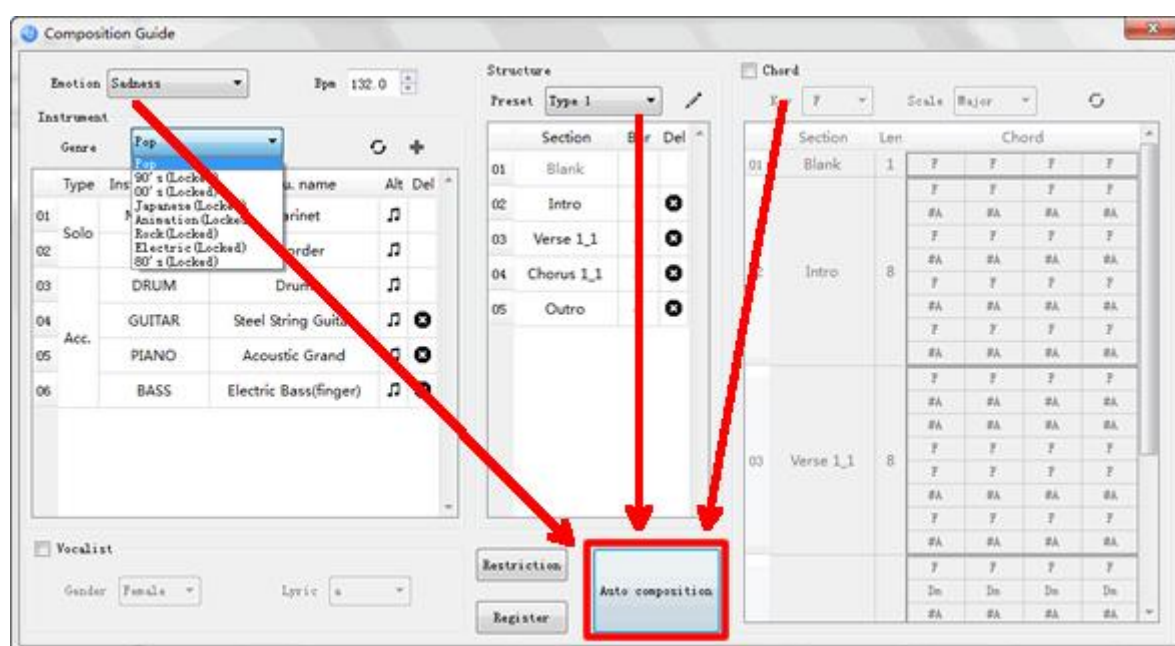
[http://www.danielbrownmusic.com/uploads/1/3/2/3/13234393/final\\_dissertation\\_final\\_edit.pdf](http://www.danielbrownmusic.com/uploads/1/3/2/3/13234393/final_dissertation_final_edit.pdf)

s. 77

Na początku każdej iteracji następuje pobranie informacji o stanie gry (emocje, osiągnięcia, przewidywany humor gracza itd.). Następnie przy wykorzystaniu posiadanej bazy współbrzmień i figur melodycznych, powstają progresje muzyczne na najbliższe kilka lub kilkanaście taktów, które tworzą zdania muzyczne i całe frazy. W ostatnim etapie zachodzą transformacje stochastyczne aby wprowadzić do powstałych fraz element twórczy, niepowtarzalny.

### 3.3.3. AthTek DigiBand

*DigiBand*<sup>40</sup> to rozwiązanie komercyjne, które umożliwia łatwe generowanie linii melodycznych dla wybranych instrumentów. Można wskazać rytm, metrum, instrumenty, styl muzyki oraz charakter (klimat emocjonalny) utworu, lub też nagrać próbkę dźwiękową i program sam ułoży pozostałe ścieżki dźwiękowe w dopasowaniu do wczytanej melodii. Jest to narzędzie reklamowane jako pomoc przy tworzeniu własnych kompozycji, gdyż wedle wyboru instrumentów i gatunku pożądanej muzyki, resztę pracy wykonuje zamiast człowieka. Jego zaletą jest także szybkie działanie, lecz niestety brak informacji o algorytmie z jakiego korzysta przy generowaniu muzyki.



Rysunek 3.14 Interfejs programu DigiBand, źródło <http://www.athtek.com/image/digiband/composition.jpg>, strona odwiedzona 1.09.2016

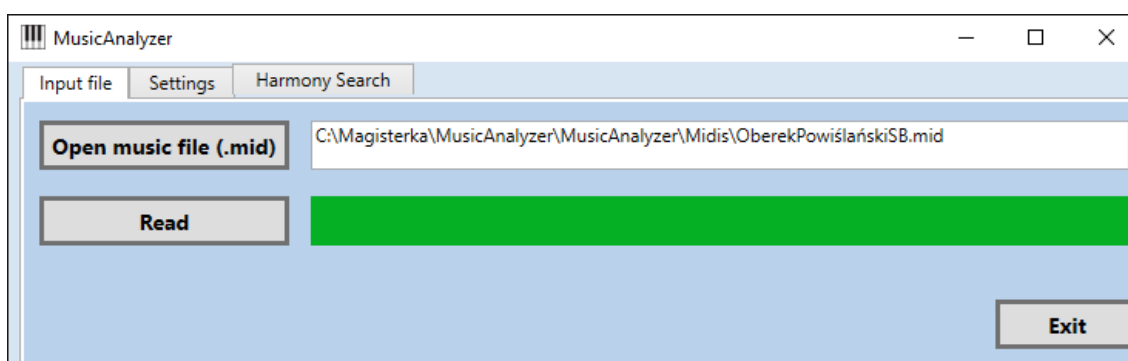
<sup>40</sup> <http://www.athtek.com/digiband/music-composition-software.html>, dostęp 1.09.2016

## 4. Opis rozwiązania

*MusicAnalyzer* to aplikacja z przeznaczeniem na komputer stacjonarny z systemem Windows, realizująca założenia projektu pracy dyplomowej. Umożliwia uruchomienie automatycznego komponowania muzyki, obrazuje i pozwala zapisać jego wyniki i ułatwia ocenę ich jakości.

### 4.1. Opis interfejsu graficznego programu *MusicAnalyzer*

- Widok początkowy – klasa *StartWindow* / zakładka *Input file*



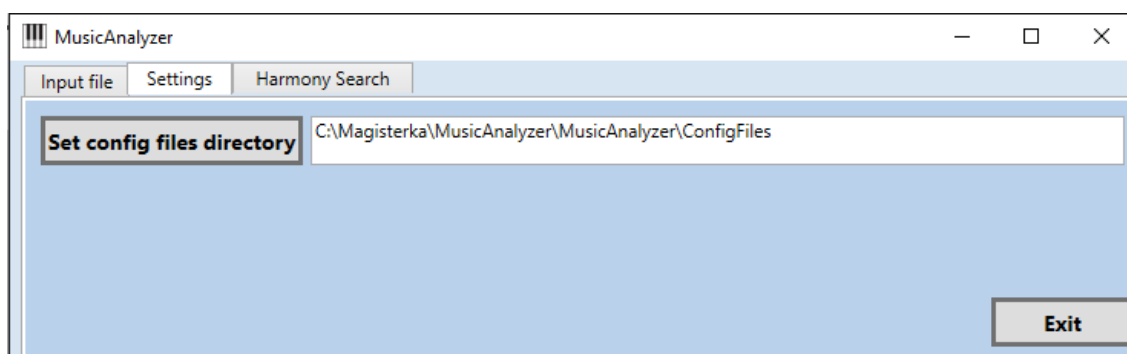
Rysunek 4.1 Widok początkowy okna *StartWindow* w programie *MusicAnalyzer*

Jest to widok początkowy po uruchomieniu aplikacji, umożliwia on wskazanie wejściowego pliku dźwiękowego w formacie *MIDI*. Na zakładce *Input file* widoczne jest pole wskazujące na adres pliku oraz pasek stanu pokazujący procentowy postęp wczytywania informacji z pliku. Dostępne są także następujące przyciski:

- *Open music file (.mid)* – po jego naciśnięciu otwiera się systemowe okno dialogowe umożliwiające wybór pliku źródłowego z linią melodyczną, dla której możliwe będzie utworzenie akompaniamentu. Możliwe jest wskazanie jedynie pliku o rozszerzeniu *.mid*. Po wybraniu pliku, w polu po prawej prezentowana jest ścieżka dostępu do pliku w systemie.
- *Read* – Wykonuje wczytywanie pliku wejściowego i przygotowuje zestaw informacji potrzebny do prawidłowej prezentacji zapisu nutowego w kolejnym oknie. Proces ten składa się z kilku kroków i po zakończeniu każdego z nich aktualizowany jest pasek postępu tego zadania. Jest ono uruchamiane w osobnym wątku, dzięki czemu nie jest blokowany interfejs. Po zakończeniu tego procesu prezentowane jest okno *PlayerWindow*, w którym przedstawiony jest zapis nutowy wczytanych ścieżek dźwiękowych.
- *Exit* – przycisk ten zamyka aplikację.



- Widok początkowy – klasa *StartWindow* / zakładka *Settings*

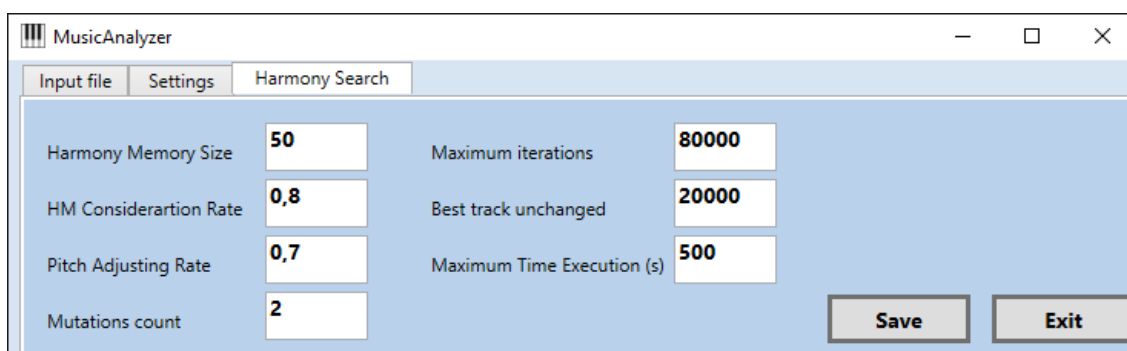


Rysunek 4.2 Widok drugiej zakładki w oknie *StartWindow* w programie *MusicAnalyzer*

Na tym widoku mamy możliwość określenia lokalizacji folderu z plikami konfiguracyjnymi. Domyślnie jest prezentowana wartość używana podczas testów rozwiązania, lecz każdorazowo jest ona dostępna do zmiany. Dostępne są następujące przyciski:

- *Set config files directory* – po jego naciśnięciu otwiera się systemowe okno dialogowe umożliwiające wybór folderu, w którym znajdują się pliki konfiguracyjne. Są one niezbędne zarówno do wyświetlenia zapisu nutowego oraz dla utworzenia akompaniamentu. Po wybraniu lokalizacji, w polu po prawej prezentowana jest pełna ścieżka dostępu w systemie.
- *Exit* – po wyborze tego przycisku aplikacja jest zamykana.

- Widok początkowy – klasa *StartWindow* / zakładka *Harmony Search*



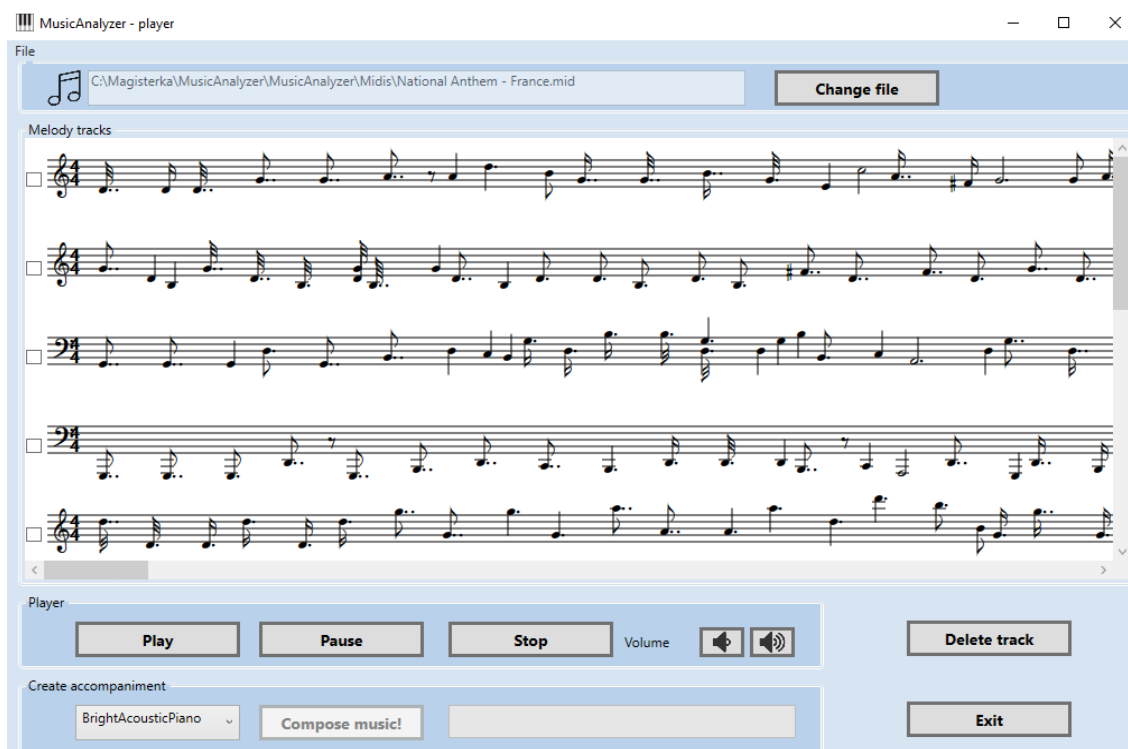
Rysunek 4.3 Widok trzeciej zakładki w oknie *StartWindow* w programie *MusicAnalyzer*

Na tej zakładce mamy możliwość ustawienia parametrów wywołania algorytmu *Harmony Search*, który zostanie użyty do wygenerowania akompaniamentu muzycznego wczytanej melodii. Wartości parametrów są walidowane, a ich znaczenie zostało opisane w tej pracy w rozdziale dotyczącym samego algorytmu<sup>41</sup>. Znajdujące się w polach wartości domyślne zostaną

<sup>41</sup> Patrz rozdział 4.5.1.

zastosowane w razie niepomyślnej walidacji po dokonaniu zmian. Aby wprowadzone wartości zostały zapisane, należy kliknąć przycisk *Save*.

- Główne okno aplikacji – klasa *PlayerWindow*

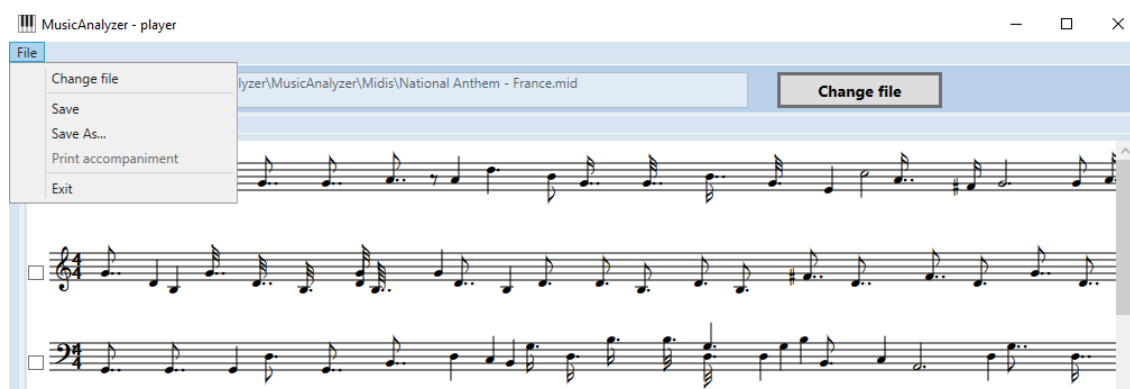


Rysunek 4.4 Widok głównego okna *PlayerWindow* w programie *MusicAnalyzer*

Jest to główne okno aplikacji spełniające podstawowe funkcje swojego przeznaczenia. Zostało podzielone na następujące panele:

- *File* – prezentowana jest tu informacja o pliku wejściowym, jego ścieżka dostępu w systemie. Poprzez wybranie przycisku *Change file* można powrócić do pierwszego okna i zmienić plik wejściowy.
- *Melody tracks* – element graficzny przedstawiający w każdym wierszu osobną ścieżkę dźwiękową z pliku wejściowego a u dołu także tę skomponowaną przez program. Każdy z wierszy zawiera dwa elementy:
  - Pole wyboru – należy je zaznaczyć, jeśli chcemy usunąć daną ścieżkę dźwiękową.
  - Element graficzny z zapisem nutowym – zawiera pięciolinię z należącymi do danej ścieżki dźwiękami. Wszystkie nuty prezentowane są w jednym wierszu a te, które nie mieszczą się na pierwotnym widoku, można obejrzeć przesuwając suwak u dołu formatki.

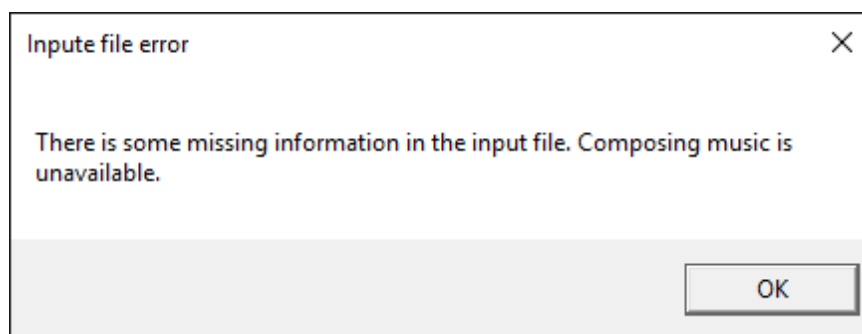
- *Player* – ten panel umożliwia odtwarzanie pliku muzycznego. Po dodaniu ścieżki akompaniamentu pozwala na wysłuchanie jej wraz z brzmieniem podstawowym. W tej grupie dostępne są następujące elementy:
    - *Play* – przycisk uruchamiania odtwarzania z ostatnio zakończonego miejsca (domyślnie jest to początek utworu).
    - *Pause* – przycisk pauzy, która umożliwia zapamiętanie miejsca przzerwania odtwarzania.
    - *Stop* – kończy odtwarzanie pliku muzycznego
    - *Volume (+/-)* – dwa przyciski pozwalające na systemowe zwiększenie bądź zmniejszenie głośności muzyki emitowanej przez domyślne urządzenie *audio*.
  - *Create accompaniment* – ten panel umożliwia utworzenie akompaniamentu do ścieżek dźwiękowych zawartych w pliku wejściowym. Zawiera następujące elementy:
    - Lista rozwijalna – lista dostępnych instrumentów *MIDI* (128 pozycji). Wybór wskazuje na instrument, którego brzmienie ma zostać użyte do powstającej ścieżki dźwiękowej. Domyślnie wskazany jest dźwięk fortepianu (*bright acoustic piano*).
    - *Compose music!* – przycisk uruchamiający proces tworzenia akompaniamentu. Postęp procentowy procesu prezentowany jest na pasku po prawej stronie. Przycisk jest dostępny, gdy wczytany plik zawiera wszystkie potrzebne informacje oraz jeśli nie uruchomiono już komponowania. Aby móc to zrobić ponownie, należy powrócić do widoku okna początkowego i na nowo wczytać plik muzyczny.
  - Pozostałe przyciski:
    - *Delete track* – naciśnięcie tego przycisku powoduje usunięcie z pliku wejściowego ścieżek dźwiękowych wybranych poprzez zaznaczenie znacznika przy każdym z zapisów nutowych.
    - *Exit* – zamyka bieżące okno.
- Menu dostępne w głównym oknie – klasa *PlayerWindow* / przycisk *File* / ...



Rysunek 4.5 Widok opcji Menu w oknie PlayerWindow w programie MusicAnalyzer

- *Change file* – ta opcja umożliwia zmianę pliku wejściowego. Zamyka bieżące okno, co przywraca widok okna początkowego, gdzie można wskazać i wczytać nowy plik.
- *Save* – zachowuje bieżącą konfigurację w pliku wejściowym, co zawierać może dodanie nowo skomponowanej ścieżki dźwiękowej lub usunięcie istniejących.
- *Save As...* - opcja pozwala na wybór nazwy pliku, do jakiego ma być zachowany stan bieżącej kompozycji.
- *Print accompaniment* – po wyborze tej opcji uruchamiane jest systemowe narzędzie drukowania. Na wydruku prezentowana będzie jedynie ścieżka akompaniamentu w podziale na wiersze i strony w formacie pionowym.
- *Exit* – zamyka bieżące okno aplikacji.

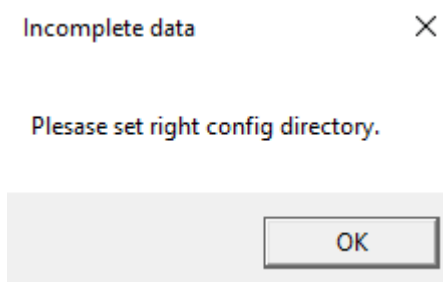
W momencie uruchomienia głównego okna aplikacji następuje sprawdzenie, czy w pliku wejściowym znajdują się wszystkie informacje potrzebne do skomponowania akompaniamentu. Program sprawdza, czy utwór posiada oznaczenie tonacji, czy ma ustalone metrum oraz czy udało mu się znaleźć jakiegokolwiek dźwięki. Jeśli choć jeden z powyższych warunków nie jest spełniony, komponowanie nie będzie możliwe i pojawi się następujący komunikat:



Rysunek 4.6 Komunikat o błędzie pliku wejściowego w programie MusicAnalyzer

#### 4.2. Informacje bazowe – pliki konfiguracyjne

Do prawidłowego działania programu potrzebny jest zestaw plików konfiguracyjnych, które są dołączone także do projektu *MusicAnalyzer*. Aplikacja wymaga wskazania lokalizacji, w której mają się one znajdować. Jeśli ich tam nie znajdzie, podczas wczytywania pliku wejściowego pokaże się komunikat:



Rysunek 4.7 Komunikat o braku plików konfiguracyjnych w *MusicAnalyzer*

Dalsze działanie programu zostanie przerwane, konieczne będzie wskazanie innej lokalizacji na komputerze zawierającej wszystkie niżej wskazane pliki:

- **basicPitches.txt** – plik zawiera 12 wierszy, w których wyszczególnione są wszystkie podstawowe dźwięki gamy (C-dur) w postaci pary nr kolejny i oznaczenie muzyczne np. „0 C”; dzięki niemu możliwe jest np. przyporządkowanie nazw dźwięków do określonej wartości zapisanej w sygnale MIDI
- **minorSequence.txt** – plik ten zawiera 7 wierszy (tyle dźwięków zawiera każda gama), a każdy z nich zawiera numer kolejny dźwięku gamy molowej z ciągu wszystkich dźwięków dostępnych na klawiaturze fortepianu w obrębie oktawy (zawartych w pliku *basicPitches.txt*); umożliwia to rozpoznanie, czy dany dźwięk należy do skali oraz z jakich dźwięków można składać akordu tonalne
- **majorSequence.txt** – ten plik zawiera analogiczne 7 wierszy, jak opisany powyżej, z tym, że jest to ciąg dźwięków dowolnej gamy durowej
- **majorChords.txt** – zawiera spis akordów (trójdźwięków) charakterystycznych dla gamy durowej:
  - tonika
  - subdominanta
  - dominanta (w skali harmoniczej)
  - trójdźwięk na 2-im stopniu gamy
  - trójdźwięk na 3-im stopniu gamy
  - trójdźwięk na 6-ym stopniu gamywidza ta pozwala stworzyć powyższe akordy odpowiednie dla każdej tonacji
- **minorChords.txt** – zawiera spis akordów (trójdźwięków) charakterystycznych dla gamy molowej:
  - wszystkie trójdźwięki wskazane dla gamy durowej

- dominanta (w skali eolskiej)
- trójdźwięk na 3-im stopniu gamy (w skali eolskiej)

### **4.3. Wczytywanie melodii wejściowej i metody określające jej cechy**

Proces wczytywania i analizowania informacji zawartych w wejściowym pliku *MIDI* w programie *MusicAnalyzer* składa się z kilku etapów (uruchamianych w osobnych wątku pracy procesora, dzięki czemu interfejs użytkownika nie ulega zablokowaniu).

#### **4.3.1. Zdekodowanie sygnałów dźwięków**

Obiekt pliku dźwiękowego przechowuje klasa *Sequence* (z biblioteki *Sanford*<sup>42</sup>), która jest kolekcją obiektów typu *Track*, które odpowiadają ścieżkom dźwiękowym, do których przypisane są sygnały dźwiękowe. Należy zatem iterować każdą z nich, aby uzyskać wszystkie dźwięki. Jednak do ścieżki dźwiękowej przypisane są nie tylko sygnały pojawienia się i wygaśnięcia dźwięku, dlatego z kolekcji obiektów *MidiEvent* należy wybrać tylko te będące typu *ChannelMessage*. Następnie jeśli mamy do czynienia z sygnałem o komendzie *NoteOn*, to jest to sygnał pojawienia się dźwięku, który dodatkowo posiada atrybuty:

- Wysokość dźwięku (możliwe wartości z zakresu  $<A0 = 21; C8 = 109>$ )
- Poziom głośności dźwięku (0 – 100)
- Umiejscowienie w czasie (ilość taktowań od początku utworu)
- Nr ścieżki dźwiękowej

Niestety w plikach *MIDI* moment zakończenia dźwięku jest przez różne urządzenia oznaczany dwojako: poprzez nadanie sygnału *NoteOff* lub sygnału *NoteOn*, którego poziom natężenia dźwięku = 0. W przypadku spełnienia jednego z powyższych warunków zapisywane są dodatkowe atrybuty wskazanej nuty:

- Oznaczenie znakowe nuty (np. C4)
- Jej odpowiednik w podstawowej skali (C)
- Oktawa
- Koniec trwania
- Długość trwania

#### **4.3.2. Wyszukanie zmian tonacji utworu**

---

<sup>42</sup> Zob. Punkt 4.7.1 niniejszej pracy

Podobnie, jak odszyfrowywanie dźwięków, wyszukiwanie zmian tonacji utworu wymaga przejrzenia wszystkich ścieżek w pliku i przypisanych doń sygnałów. W tym przypadku należy jednak odnaleźć obiekty *MidiEvent*<sup>43</sup> będące typu *MetaMessage*, których atrybut *MetaType* = *KeySignature*. Tak powstają instancje klasy *Tonation*, które następnie są uzupełniane między innymi o atrybuty:

- *Offset* – przesunięcie tonacji względem skali C-dur
- Zbiór dźwięków gamy należących do tej skali
- Dźwięki wszystkich akordów należących do tej skali

Dodatkowo, po znalezieniu wszystkich zmian tonacji, dla każdej z nich ustalany jest moment jej początku i końca (ilość taktów od początku utworu).

#### **4.3.3. Wyszukiwanie zmian metrum**

Wyznaczanie metrum i jego zmian odbywa się analogicznie jak dla tonacji, z tą różnicą, że atrybut *MetaType* musi przyjmować wartość *TimeSignature*. Po znalezieniu wszystkich takich zdarzeń, zapisywane jest także momenty początku i końca okresu ich obowiązywania.

#### **4.3.4. Ustalenie prawidłowej tonacji utworu**

Niestety tonacja utworu odczytana z pliku *MIDI* przy pomocy narzędzi z biblioteki *Sanford.Multimedia*<sup>44</sup>, nie daje nam gwarancji, że jest prawidłowa. Wynika to z faktu, iż w pliku zapisana jest wartość przesunięcia danej tonacji względem tonacji bazowej (C-dur bądź a-moll). Dlatego, aby odróżnić tonację molową od durowej, która posiada te same znaki przy kluczu, musimy wykonać analizę występowania charakterystycznych dźwięków skali.

- a) Dla dowolnej pary tonacji dur – moll, posiadających te same znaki przy kluczu, większość dźwięków występując w zapisie nutowym jest taka sama<sup>45</sup>. Istotna różnica może tkwić w siódmym podwyższonym dźwięku w skali molowej harmoniczej, który w takim przypadku nie ma swojego odpowiednika w skali durowej. Kompozytorzy stosują ten zabieg, aby uwydatnić brzmienie dominanty, która wówczas buduje napięcie w utworze dążące do rozwiązania na akordzie toniki. Przykładem może być pieśń S. Moniuszki „Kozak”, w której przed zakończeniem każdego wersu w zwrotce występuje akord dominanty z podwyższonym VII stopniem gamy i następnie rozwiązuje się on na tonikę.

---

<sup>43</sup> Zob. Punkt 4.7.1 niniejszej pracy

<sup>44</sup> Zob. Punkt 4.7.1 niniejszej pracy

<sup>45</sup> *Zanim zaczniesz grać na...*, dz. cyt., s.19



Rysunek 4.8 Fragment zapisu nutowego pieśni "Kozak" S. Moniuszki z akompaniamentem pianina (c-moll).

Oczywiście, ten zabieg nie jest obowiązkowy, lecz występuje w ogromnej większości kompozycji klasycznych<sup>46</sup>, co w porównaniu z faktem, iż odpowiadający mu piąty dźwięk w skali durowej praktycznie nigdy nie jest podwyższany, umożliwia dość celnie określić, czy dany utwór jest w skali durowej czy molowej. Skonstruowano zatem warunek realizujący to założenie:

Jeśli ilość wystąpień VII stopnia  $\leq (2 * \text{ilość wystąpień podwyższonego VII st.})$

To jest to gama molowa

Inaczej jest to gama durowa

- b) Drugą cechą utworu poddawaną analizie, jest statystyka akordów na początków taktów, czyli w miejscach najistotniejszych dla budowania rytmu i układu harmonicznego. Są to węzły, które osadzają rytm, występują na nich dłuższe wartości rytmiczne i to one w dużej mierze charakteryzują tonację, w jakiej napisany zostały badany utwór<sup>47</sup>. Dlatego równorzędnym warunkiem dla analizy występowania podwyższonego VII stopnia gamy, jest określenie, czy częściej na początku taktu występuje akord toniki w tonacji dur czy moll:

Jeśli ilość wystąpień toniki durowej na początku taktu  $\leq$  ilość wystąpień toniki molowej na początku taktu

To jest to gama molowa

<sup>46</sup> A. Poszowski, *Harmonia systemu tonalnego...*, dz. cyt., s. 33

<sup>47</sup> Piotr Kałużny, *Skale muzyczne...*, dz. cyt., s. 23



Inaczej jest to gama durowa

Oba opisane powyżej warunki zostały połączone alternatywą, zatem aby gamę uznać za molową, wystarczy spełnić tylko jeden z nich.

#### 4.3.5. Ustalenie prawidłowych oznaczeń dźwięków

Kolejną niejednoznacznością, z jaką należy się zmierzyć korzystając z plików *MIDI*, jest fakt, iż posiadając jedynie bezwzględną wysokość każdego z dźwięków, nie jesteśmy w stanie od razu zaprezentować ich na pięciolinii w zapisie nutowym. Dla przykładu, dźwięk o wysokości 49 (w formacie *MIDI*) może być jednocześnie oznaczony jako Cis4 i Des4<sup>48</sup>. Jest to uzależnione od tonacji utworu w którym te dźwięki występują.



Rysunek 4.9 Tonacje i znaki przy kluczu wiolinowym, źródło

[http://blog.gitarius.pl/kurs\\_gitarowy/chro4.png](http://blog.gitarius.pl/kurs_gitarowy/chro4.png)

Jeśli wskazana tonacja przy kluczu posiada krzyżyki, wówczas dźwięk ten oznaczylibyśmy jako Cis4, w przeciwnym razie (posiada bemole lub nie posiada znaków przy kluczu), to będzie on oznaczony jako Des4. Ma to znaczenie przede wszystkim pod kątem prawidłowej prezentacji zapisu nutowego, lecz także pozwala uniknąć problemów z porównywaniem dźwięków w innych etapach prowadzonych analiz.

#### 4.3.6. Ustalenie wartości rytmicznych wszystkich nut

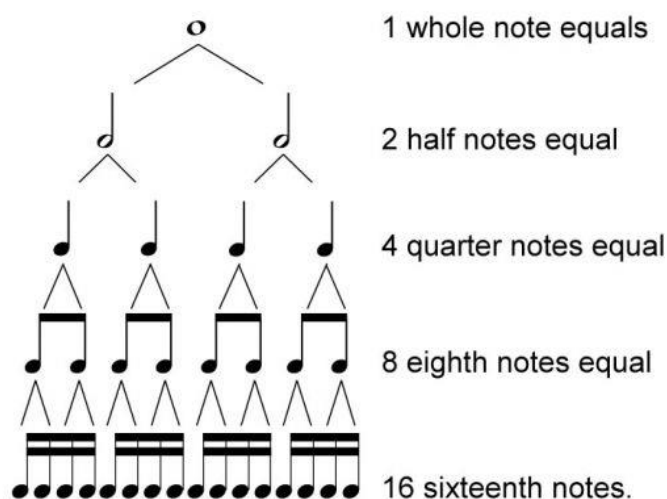
Ostatnim etapem uzupełniania atrybutów wczytanych nut jest prawidłowe dopasowanie wartości rytmicznych względem długości ich trwania. Jest to możliwe dzięki uzyskaniu we wcześniejszych etapach informacji o metrum utworu oraz zapisanej w pliku wartości *Division* – ilości taktowań zegara w ramach jednej ćwierćnuty<sup>49</sup>. Mając tą wiedzę jesteśmy w stanie przypisać wartości rytmiczne dźwiękom w zależności od długości ich trwania. Ten krok jest konieczny dla prawidłowego zaprezentowania zapisu nutowego, dlatego też *MusicAnalyzer*

<sup>48</sup> Zanim zaczniesz grać na..., dz. cyt., s.23

<sup>49</sup> J. P. Bello, *MIDI Code*, [https://www.nyu.edu/classes/bello/FMT\\_files/9\\_MIDI\\_code.pdf](https://www.nyu.edu/classes/bello/FMT_files/9_MIDI_code.pdf) (dostęp 14.09.2016)

przechowuje ten atrybut w formie typu wyliczeniowego

*PSAMControlLibrary.MusicalSymbolDuration*<sup>50</sup>, który jest rozpoznawany przez komponent graficzny (*IncipitViewerWPF*) prezentującego nuty na pięciolinii.



























Rysunek 4.10 Podział wartości rytmicznych, źródło  
[http://whittier.mpls.k12.mn.us/uploads/rhythm\\_tree\\_1.jpg](http://whittier.mpls.k12.mn.us/uploads/rhythm_tree_1.jpg)

Powyższy schemat przedstawia jedynie podstawowe wartości, których długość jest pewną potęgą dwójki, co oczywiście nie wystarcza, by odpowiednio zaprezentować wszystkie nuty. Dlatego zapis nutowy umożliwia przedłużanie wskazanej wartości o połowę poprzez dodanie obok kropki<sup>51</sup>. Możliwe jest także dodawanie kolejnych kropek, wówczas każda z nich przedłuża o dwukrotnie mniejszą wartość ( $1/4$ ,  $1/8$  bazowej wartości rytmicznej itd.). Ilustruje to poniższa tabela.

<sup>50</sup> Zob. Punkt 4.7.2 niniejszej pracy

<sup>51</sup> *Zanim zaczniesz grać na...*, dz. cyt., s.27

*Tabela 4.1 Wartości rytmiczne przedłużane, źródło*  
<https://musictheorysite.files.wordpress.com/2015/06/dotted-value.png>

Notes	Name	Value	Rests	Name	Value
	Dotted minim/dotted half note	 + 		Dotted minim rest/dotted half note rest	 + 
	Dotted crotchet/dotted quarter note	 + 		Dotted crotchet rest/dotted quarter note rest	 + 
	Dotted quaver/dotted eighth note	 + 		Dotted quaver rest/dotted eighth note rest	 + 
	Dotted semiquaver/dotted sixteenth note	 + 		Dotted semiquaver rest/dotted sixteenth note rest	 + 

W programie *MusicAnalyzer* w procesie dopasowywania wartości rytmicznych przyjęto założenie, że nuta może mieć maksymalnie dwie kropki przedłużające jej wartość, gdyż w praktyce nie spotyka się ich w większej ilości. Występuje tu jednak dodatkowa trudność w związku z zapisem dźwięków w formacie *MIDI*. Mimo, że w zapisie nutowym dwa dźwięki występują bezpośrednio po sobie, to jednak, aby nie powodować wrażenia nachodzenia ich na siebie oraz by odtwarzacz pliku prawidłowo wyłączył pierwszy dźwięk i włączył kolejny, wszystkie dźwięki zazwyczaj są trochę krótsze (o ok. 10-15% czasu)<sup>52</sup>. Powoduje to problem matematyczny, gdyż wskazane wartości należy odpowiednio zaokrąglić w górę, lecz jednocześnie odróżnić od siebie różne wartości o zbliżonych do siebie długościach (np. półnutę od ćwierćnutę z dwiema kropkami).

#### 4.4. Prezentacja graficzna zapisu nutowego

Po ustaleniu wszystkich atrybutów nut, zmian tonacji i metrum, należy odpowiednio przygotować każdą z pięciolinii. Biblioteka *PSAMWPFFControlLibrary*<sup>53</sup> umożliwia prezentowanie zapisu nutowego w elemencie graficznym typu *IncipitViewerWPF*. Jego interfejs wymaga podawania po kolei wszystkich oznaczeń, które mają się pokazać na pięciolinii, a zatem nie tylko nut, ale też klucza, oznaczenia metrum czy kresek taktowych. Do tego celu służy przeciążona metoda:

```
PSAMWPFFControlLibrary.IncipientViewerWPF.AddMusicalSymbol(MusicalSymbol symbol)
```

<sup>52</sup> J. P. Bello, *MIDI Code*, [https://www.nyu.edu/classes/bello/FMT\\_files/9\\_MIDI\\_code.pdf](https://www.nyu.edu/classes/bello/FMT_files/9_MIDI_code.pdf) (dostęp 14.09.2016)

<sup>53</sup> Zob. Punkt 4.7.3 niniejszej pracy

Umożliwia dodanie do zapisu dowolnych obiektów, których klasy dziedziczą po klasie *MusicalSymbol*.

*MusicAnalyzer* prezentuje nuty przypisane do osobnych ścieżek dźwiękowych na oddzielnych pięcioliniiach. W tym celu, dla każdego kanału należy wskazać w jakim kluczu przedstawić zapis nutowy (wiolinowym czy basowym).



*Rysunek 4.11 Fragment partii tenora z chóralnej aranżacji utworu ludowego "Oberek powiślański"*

Powyższy przykład dobrze ilustruje problem z wyborem klucza dla wskazanej melodii. Tessitura głosu tenorowego jest często zbyt wysoka do zapisu w kluczu basowym, natomiast bywa też za nisko, jak na klucz wiolinowy (przykład na rysunku)<sup>54</sup>. W obu przypadkach łączy się to z koniecznością dorysowywania dodatkowych linii powyżej bądź poniżej pięciolinii. Prowadzona jest zatem analiza wysokości wszystkich nut należących do każdej ścieżki dźwiękowej, aby określić, który z kluczy pozwala zaprezentować je bardziej przejrzysto. Program zlicza nuty, które występują powyżej i poniżej wartości granicznej, za którą przyjęto dźwięk C4 – dźwięk równie odległy od obu pięciolinii. Jeśli więcej jest dźwięków wyższych lub równych C4, to zapis będzie prowadzony w kluczu wiolinowym, w przeciwnym wypadku - w kluczu basowym.

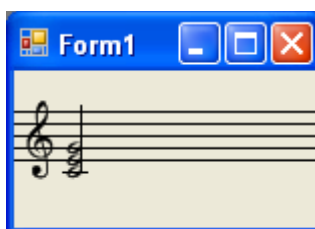
Po ustaleniu klucza danej pięciolinii, program iteruje po wszystkich nutach należących do wskazanej ścieżki dźwiękowej. Zanim doda ją do kolekcji elementów należących do pięciolinii, każdorazowo sprawdza, czy w momencie pojawienia się kolejnej nuty nie jest także przypisany sygnał zmiany tonacji lub metrum. W takim przypadku, najpierw należy wstawić znaki przy kluczu bądź nowe metrum a dopiero później nutę<sup>55</sup>. Dodatkowo następuje sprawdzanie, czy na ten moment nie przypada koniec taktu. Wówczas należy postawić znak kreski taktowej. Ostatnim elementem, który może poprzedzać nutę jest znak pauzy wynikający z faktu, iż poprzednia nuta

<sup>54</sup> A. Długosz, *Kształcenie głosu. Śpiew solowy i chóralny. Jak uczyć się i uczyć innych*, nakład własny, Warszawa 1936, s. 16

<sup>55</sup> Irena Poniatowska, *Dzieło muzyczne...*, dz. cyt., s. 144

skończyła się wcześniej. Jeśli zbiega się to dodatkowo z momentem stawiania kreski taktowej, należy sprawdzić, czy pauza należy jeszcze do poprzedniego taktu, czy już do następnego i umieścić znaki w odpowiedniej kolejności. Program każdorazowo wylicza odpowiednią długość przerwy i przyporządkowuje jej wartość rytmiczną. W programie *MusicAnalyzer* przyjęto uproszczenie, iż wstawiane pauzy będą miały swoje nominalne wartości rytmiczne i nie będą przedłużane o połowę bądź jedną czwartą.

Gdy na pięciolinii należy jednak umieścić nutę, program uzupełnia jej wcześniej wyliczone atrybuty jeszcze o bezpośrednie cechy widoku. Należy do nich połączenie nut zaczynających się w tym samym momencie i należących do tego samego kanału dźwiękowego w pionowy akord. Tę zależność i jej prezentację przedstawia poniższa przykładowa ilustracja:



*Rysunek 4.12 Trójdźwięk zapisany na pięciolinii, źródło <http://www.codeproject.com/KB/miscctrl/psamcontrollibrary/examplechords.png>, strona odwiedzona 1.09.2016*

Aby nuta była w ten sposób zaprezentowana, musi otrzymać wartość wskaźnika *isChordElement = true*. Konstruowanie akordów odbywa się poprzez porównanie aktualnej nuty z poprzednią. Jeśli obie zaczynają się w tym samym czasie, to należy im ustawić powyższą flagę, w przeciwnym razie przyjmuje ona domyślną wartość *false*.

Ostatnim atrybutem nuty pozostającym do ustalenia przed wygenerowaniem jej widoku jest kierunek ogonka nuty, jeśli dana wartość rytmiczna taki posiada. Dotyczy to zatem wszystkich nut za wyjątkiem całej nuty. Możliwe różnice w prezentacji przedstawia poniższy rysunek:



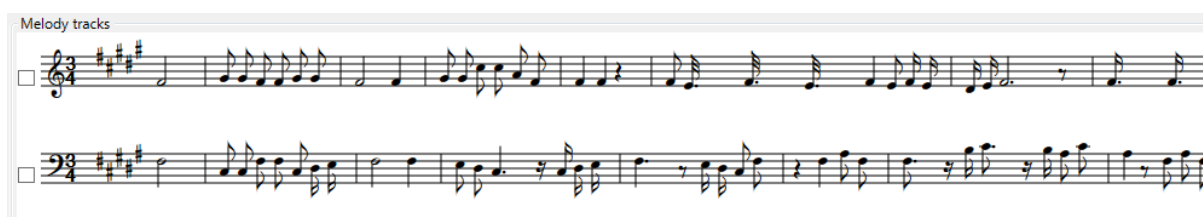
*Rysunek 4.13 Fragment zapisu nutowego z nutami o różnym kierunku*

Widać wyraźnie, że nuta o niższej wysokości dźwięku prezentowana w dolnej części pięciolinii jest prezentowana bardziej przystępnie, gdy jej ogonek jest skierowany do góry, a dla wyższych nut analogicznie w dół. *MusicAnalyzer* dla każdej nuty określa ten kierunek w zależności od wysokości nuty i klucza pięciolinii, na jakiej się znajduje<sup>56</sup>. Nuty z ogonkami 'w dół' w kluczu wiolinowym to od H4 i wyżej, a niższe mają ogonek 'do góry'. W kluczu basowym ogonek 'w dół' mają nuty od D3 i wyżej, a niższe 'do góry'. Za przechowanie tej informacji na potrzeby

<sup>56</sup> Irena Poniatowska, *Dzieło muzyczne...*, dz. cyt., s. 103

wyświetlenia nuty w obiekcie *IncipitViewerWPF* odpowiada typ wyliczeniowy *PSAMControlLibrary.NoteStemDirection*. W przyjętym rozwiązaniu zastosowano także powiązanie nut należących do tego samego akordu, które są prezentowane w pionie. W takim przypadku, wiążąca jest pozycja najniższej nuty w akordzie, ona determinuje kierunek ogonka, a pozostałe prezentowane są z tym samym kierunkiem.

Po ustaleniu opisanych wyżej atrybutów związanych z prezentacją zapisu nutowego, obiekt klasy *IncipitViewerWPF* wyświetla wszystkie elementy zgodnie z kolejnością podania ich na wejściu. Biblioteka *PSAMControlLibrary* umożliwia też wiązanie sąsiadujących ze sobą nut poprzez określenie ich atrybutu w postaci typu wyliczeniowego *PSAMControlLibrary.NoteBeamType*, lecz w tym projekcie to rozwiązanie nie jest stosowane. Jeśli nuty nie są częścią akordu granego jednocześnie, każdorazowo prezentowane są osobno.



Rysunek 4.14 Przykładowy zapis nutowy dwóch głosów w programie *MusicAnalyzer*

#### 4.5. Algorytm tworzenia akompaniamentu

Do znalezienia możliwie najlepszego akompaniamentu melodii wejściowej używany jest algorytm *Harmony Search*, który poprzedzony jest procesem wyszukiwania współbrzmień i akcentów rytmicznych w pliku wejściowym. Po ustaleniu szeregu akordów, należy na ich podstawie stworzyć nuty ze wszystkimi atrybutami potrzebnymi do ich prezentacji oraz zapisu do nowej ścieżki dźwiękowej. Wszystkie te kroki wykonywane są w osobnym wątku procesora, co zapobiega blokowaniu interakcji użytkownika z aplikacją.

##### 4.5.1. *Harmony Search*

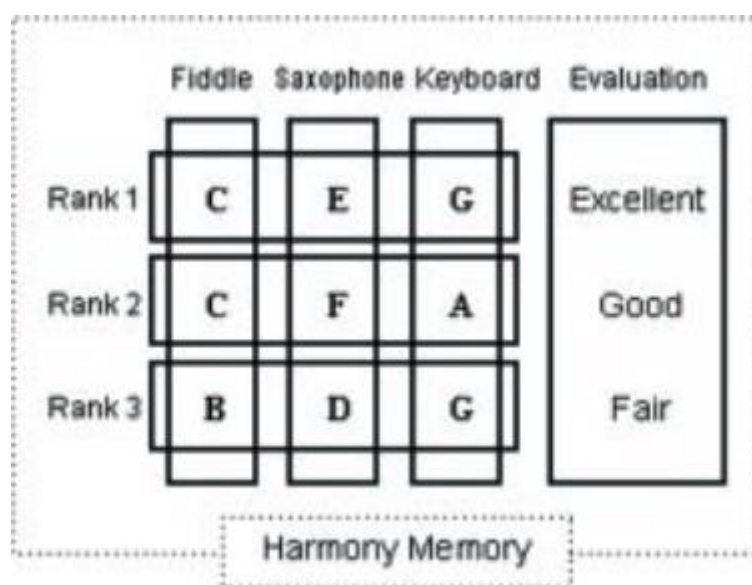
*Harmony Search* to heurystyczny algorytm optymalizacyjny z obszaru rozwiązań sztucznej inteligencji. Jego koncepcja pojawiła się całkiem niedawno, gdyż została przedstawiona w 2001 roku przez Joong Hoon Kim i Zong Woo Geem z uniwersytetu w Seulu<sup>57</sup>. Tak jak wiele innych algorytmów z tej dziedziny jest inspirowany naturalnym fenomenem, w tym przypadku harmonią muzyczną. Jest to algorytm optymalizacyjny, zatem polega on na ciągłym udoskonalaniu swojego najlepszego rozwiązania aż do osiągnięcia określonych warunków

---

<sup>57</sup> Z. W. Geem, J. H. Kim, *A New Heuristic Optimization Algorithm: Harmony Search*, 2001

końcowych. Nie daje niestety gwarancji osiągnięcia rozwiązania optymalnego, lecz w wielu złożonych zagadnieniach pozwala się do niego zbliżyć. Jego zastosowaniem mogą być zatem liczne niedeterministyczne problemy wielomianowe, które nie posiadają znanych optymalnych rozwiązań<sup>58</sup>. Należą do nich np. problem komiwojażera, znajdowanie globalnego minimum funkcji wielu zmiennych, generowanie krzyżówek czy planu zajęć dla szkoły.

Poszukiwana w ramach algorytmu optymalna harmonia jest imitacją sposobu tworzenia harmonii przez kilku muzyków, którzy jednocześnie improwizują melodie na swoich instrumentach<sup>59</sup>. Złożenie ich wspólnego brzmienia jest finalnym efektem, którego poprawa jest głównym zadaniem algorytmu. Bazuje on zatem na pamięci wcześniej wygenerowanych udanych współbrzmień i na ich podstawie stara się tworzyć kolejne. Przykładowa pamięć rozwiązań i ich ocena została przedstawiona na poniższym rysunku.



Rysunek 4.15 Struktura pamięci rozwiązań (Harmony memory), źródło:  
[http://s3.amazonaws.com/academia.edu.documents/2309202/j\\_2001\\_simulation.pdf](http://s3.amazonaws.com/academia.edu.documents/2309202/j_2001_simulation.pdf)

Istotne jest zatem wskazanie warunków funkcji oceny, która określa jakość wybranego rozwiązania. W teorii jest to zestaw zasad muzycznych, których spełnienie prowadzi do uzyskania doskonale zgranego brzmienia kilku instrumentów. W praktyce, podobnie jak w wielu innych algorytmach tej klasy, polega on na nadawaniu karnych punktów za odstępstwa od zamierzonych rezultatów. Najlepsze rozwiązanie to te, które posiada ich najmniej. Określenie jej warunków wpływa na końcowe rozwiązanie, gdyż w wyniku obliczeń faworyzowane będą rozwiązania najlepiej spełniające wskazane kryteria.

<sup>58</sup> M. G.H. Omran, , M. Mahdavi, *Global-best harmony search*, Applied Mathematics and Computation (2007)

<sup>59</sup> E. Szlachcic, *Teoria i metody optymalizacji*,  
[http://staff.iiar.pwr.wroc.pl/ewa.szlachcic/materialy%20dydaktyczne/air\\_studia\\_2\\_stopnia/12w\\_timo\\_optymalizacja\\_globalna.pdf](http://staff.iiar.pwr.wroc.pl/ewa.szlachcic/materialy%20dydaktyczne/air_studia_2_stopnia/12w_timo_optymalizacja_globalna.pdf), dostęp 20.09.2016

Schemat działania algorytmu można opisać w kilku powtarzających się krokach<sup>60</sup>:

- Krok 1: Wygenerowanie rozwiązań początkowych – wypełnienie pamięci harmonii, gdzie jej liczność określa parametr *HMS* (*Harmony Memory Size*). Dla każdego układu współbrzmień należy obliczyć jakość jego harmonii a następnie posortować rosnąco względem niego zawartość pamięci.
- Krok 2: Jeśli osiągnięto warunki stopu, to zwróć najlepsze rozwiązanie w pamięci. W przeciwnym wypadku, wskaż rozwiązanie do zmiany:
  - Z prawdopodobieństwem określonym przez parametr *HMCR* (*Harmony Memory Consideration Rate*) wybierz losowe rozwiązanie z pamięci, lub w przeciwnym razie stwórz nowe.
- Krok 3: Jeśli badane rozwiązanie zostało wybrane z pamięci, to przeprowadź improwizację w badanym rozwiązaniu zgodnie z prawdopodobieństwem określonym przez parametr *PAR* (*Pitch Adjusting Rate*)<sup>61</sup>. Ilość losowych zmian w wybranym rozwiązaniu określa parametr *Delta*.
- Krok 4: Zaktualizuj zawartość pamięci rozwiązań.:
  - Oblicz jakość powstałego w iteracji rozwiązania.
  - Jeśli badane rozwiązanie nie znajduje się w pamięci, sprawdź, czy jest lepsze od ostatniego w zestawie. Jeśli tak, to wstaw utworzone współbrzmienie w jego miejsce.
  - Posortuj rosnąco zawartość pamięci pod kątem jakości współbrzmień i przejdź do kroku 2.

Warunki końcowe algorytmu zależą od zastosowania algorytmu. Zazwyczaj wskazują osiągnięcie bariery zasobów do wykorzystania, przekroczenie limitu czasu, bądź brak progresu najlepszego wyniku od wielu iteracji.

Działanie algorytmu *HS* jest podobne do popularnego algorytmu genetycznego, ponieważ bazuje także na bazie rozwiązań, które są mutowane i w wyniku selekcji naturalnej najgorsze z nich są odrzucane w kolejnych iteracjach. Stąd też krytyka tego algorytmu ze strony wielu naukowców twierdzących, że nie jest on żadną innowacją, a jedynie wariacją algorytmu genetycznego<sup>62</sup>. Jednakże algorytm *Harmony Search* może dawać lepsze rozwiązania w bardzo złożonych problemach, w których trudne jest ustalenie zróżnicowanych rozwiązań początkowych. W wielu przypadkach także lepiej unika utknięcia w minimum lokalnym, gdyż kompletnie nowe rozwiązania w dalszych iteracjach pojawiają się nie w wyniku krzyżowania, lecz dzięki

---

<sup>60</sup> K. S. Lee, Z. W. Geem, *A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice*, Computational Methods Applied Mechanics Engineering, 194 (2005)

<sup>61</sup> M. Mahdavi, M. Fesanghary, E. Damangir, *An improved harmony search algorithm for solving optimization problems*, Applied Mathematics and Computation 188 (2007)

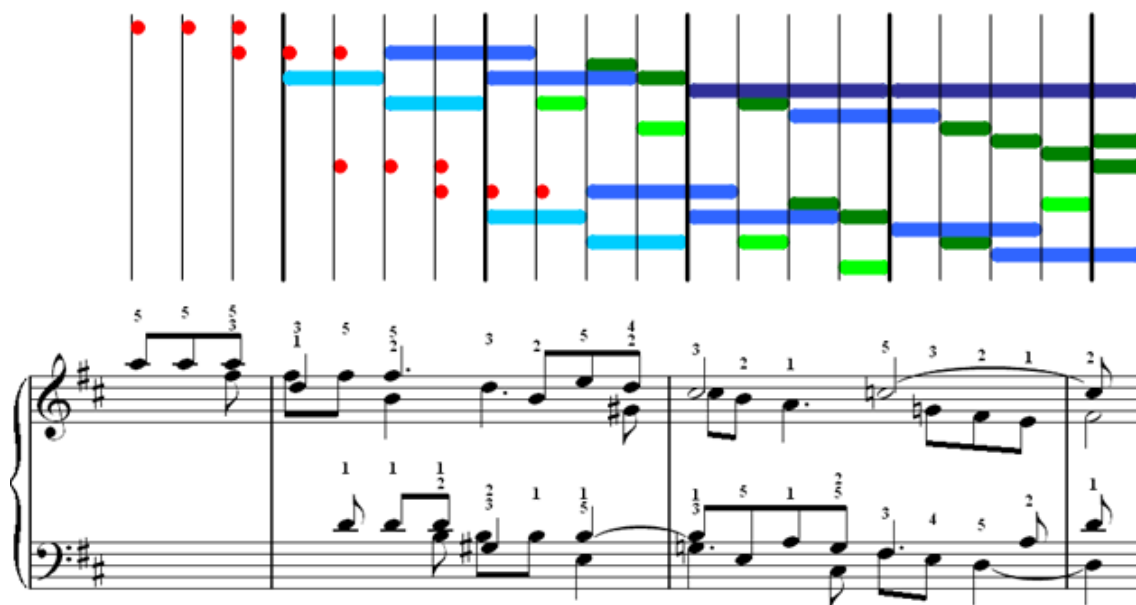
<sup>62</sup> D. Weyland, *A Rigorous Analysis of the Harmony Search Algorithm: How the Research Community can be Misled by a "Novel" Methodology*. International Journal of Applied Metaheuristic Computing. 1 (2010), s.50–60



randomizowanej funkcji generującej je w całości<sup>63</sup>. Kluczem do poprawnego działania algorytmu jest zatem trafne zamodelowanie jego działań w wybranym zastosowaniu, poprawna parametryzacja algorytmu i prawidłowe określenie funkcji oceny.

#### 4.5.2. Wyszukanie współbrzmień wejściowych i ustalenie ich atrybutów

Przed przystąpieniem do układania akompaniamentu, należy wpierw przekształcić ścieżki wejściowe w ciąg akordów, które tworzą jednocześnie brzmiące dźwięki. W programie *MusicAnalyzer* proces ten składa się z dwóch etapów. Najpierw należy wskazać wszystkie momenty, w których pojawiają się nuty z dowolnej ścieżki wejściowej. W ten sposób uzyskamy wszystkie możliwe zmiany współbrzmień. Program został przystosowany do wczytywania dowolnej ilości ścieżek z pliku wejściowego. Ograniczeniem technicznym są własności standardu *MIDI* – 16 ścieżek dźwiękowych. Kolejnym krokiem jest określenie dla każdej nuty w ilu współbrzmieniach bierze udział. Dla przykładu, długa nuta z pierwszej ścieżki dźwiękowej może pokrywać się z nutami należącymi do innej ścieżki dźwiękowej, które pojawiają się później. Każde pojawienie się nowej nuty lub koniec brzmienia wcześniejszej, skutkuje wybrzmiewaniem innego akordu. To zjawisko ilustruje poniższy rysunek przedstawiający rozkład trwających jednocześnie nut w utworze polifonicznym (fudze).



Rysunek 4.16 Jednoczesne trwanie dźwięków, źródło: <http://www.musanim.com/fugueplaying/fugueplaying.gif>

Program zapamiętuje wszystkie współbrzmienia w danym pliku wejściowym i to one będą podstawą do dopasowywania nowych akordów akompaniamentu. Po ustaleniu szeregu akordów wejściowych, każdy z nich ma przypisywaną długość swojego trwania, co umożliwi w późniejszej fazie utworzenie akompaniamentu o odpowiadającej im długości.

<sup>63</sup> Z. W. Geem, J. H. Kim, *A New Heuristic...*, dz. cyt.

Kolejną czynnością przeprowadzanej analizy pliku wejściowego jest skategoryzowanie szeregu współbrzmień jako funkcje harmoniczne w danej tonacji. Pojedyncze lub podwójne dźwięki nie reprezentują żadnej z nich, zatem atrybut *ScaleChordType* przyjmuje wówczas wartość *Other*. Dla trójdźwięków natomiast możliwe jest przyporządkowanie w pierwszej kolejności akordów triady harmonicznej (tonika, subdominanta lub dominanta) a następnie pozostałych (trójdźwięk drugiego, trzeciego i szóstego stopnia gamy). Za to przyporządkowanie odpowiada funkcja:

*public Match MusicIntelligence.matchChords(Chord a, Chord b);*

Porównuje ona dźwięki, z których składa się akord z brzmienia pliku wejściowego z akordem funkcji harmonicznej w danej gamie. Zlicza te dźwięki, które nie występują jednocześnie w obu akordach. Aby uznać akord za funkcję danego typu, dopuszczalne jest posiadanie jednego dźwięku niezgodnego z trójdźwiękiem wzorcowym. Dopasowania funkcji harmonicznych są uszeregowane w kolejności od najbardziej znaczących akordów gamy do najmniej istotnych i zatrzymywany jest po znalezieniu pierwszego dopasowania.

Następnym etapem jest dopasowanie akcentów rytmicznych w takcie do metrum obowiązującego w danym fragmencie utworu. Każde dzieło, w całości lub w podziale na fragmenty, posiada swój określony rytm. Charakteryzuje się on tym, że w określonych odstępach czasu pojawiają się nuty mocniej osadzone, które z kolei przeplatają się z nutami słabiej osadzonymi rytmicznie<sup>64</sup>. Nadaje to utworowi płynność, uwydatnia akcenty i zapewnia cykliczność przeplatającego się rytmu<sup>65</sup>. Najmniejszą zamkniętą całością, w ramach której odbywa się cykl rozkładu akcentów jest jeden takt.

Rytm jest zatem silnie powiązany z metrum, miarą taktowania. Kompozycje, w których użyto dwóch miar w takcie to często utwory szybkie, energiczne. Przy taktowaniu po trzy miary mamy wrażenie taneczności dzieła, natomiast przy czterech miarach mamy do czynienia z utworem stałym, nieśpiesznym<sup>66</sup>. Jednak w danym takcie może występować dużo więcej dźwięków pojawiających się w innych momentach, które mają większe lub mniejsze znaczenie pod kątem rytmicznym. W programie *MusicAnalyzer* dla określenia istotności danego momentu w takcie zastosowano typ wyliczeniowy *MeasureBeats* przyjmujący wartości:

- *Begin* = 4 (najsilniejszy akcent - na początku taktu),
- *Strong* = 2 (silny akcent na pozostałych miarach w takcie),
- *Weak* = 1 (słabe części taktu),
- *Default* = 0 (Wartość stosowana dla pozostałych, miar nieistotnych dla osadzenia rytmu w utworze).

---

<sup>64</sup> W. Rudziński, *Muzyka dla wszystkich : z licznymi przykładami, rysunkami i tablicami*, PWM, Kraków 1948, s. 48

<sup>65</sup> Franciszek Wesolowski, *Zasady muzyki : podręcznik dla średnich szkół muzycznych*, PWM, Kraków 1980, s. 22

<sup>66</sup> W. Rudziński, *Muzyka dla wszystkich ...*, dz. cyt., s. 43

W zależności od stosowanego metrum, rozkład akcentów będzie inny. Ponadto, utwory posługujące się tym samym taktowaniem mogą posiadać swój własny, inny rytm (szczególnie melodia prowadząca), dlatego też nie można z góry określić, jaki rozkład akcentów jest najlepszy dla danego utworu. Jednak istnieje kilka reguł, które mają zastosowanie w licznych kompozycjach, dzięki czemu na tym etapie można zaproponować wstępny rozkład akcentów i dostosować go do wartości rytmicznych melodii głównej. Do najbardziej popularnych prawidłowości układu rytmicznego utworu należą:

- Akcent na początku taktu jest najbardziej znaczący<sup>67</sup>.
- Jeśli utwór ma metrum parzyste, to akcent w połowie taktu jest drugim najsilniejszym osadzeniem rytmu w takcie<sup>68</sup>.
- Przy rytmie podzielnym na trzy, wszystkie główne miary są niemal tak samo istotne<sup>69</sup>.
- Słabe części taktu to najczęściej niepełne wartości rytmiczne.

W programie *MusicAnalyzer* zastosowano domyślny układ akcentów w zależności od metrum obowiązującego w danym fragmencie utworu. Jest on możliwy do zmiany w ramach dopasowywania akordów akompaniamentu, natomiast reguły pierwotnego dopasowania przedstawiają się następująco:

Metrum	Rozkład akcentów
$\frac{2}{2}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Strong</i> };
$\frac{3}{2}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Strong</i> }
$\frac{2}{4}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Strong</i> }
$\frac{4}{2}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Weak</i> }
$\frac{3}{4}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Strong</i> }
$\frac{4}{4}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Weak</i> }
$\frac{3}{8}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Strong</i> }
$\frac{6}{4}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Weak</i> }
$\frac{4}{8}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Weak</i> }

<sup>67</sup> Franciszek Wesołowski, *Zasady muzyki ...*, dz. cyt., s. 36

<sup>68</sup> W. Rudziński, *Muzyka dla wszystkich ...*, dz. cyt., s. 59

<sup>69</sup> Franciszek Wesołowski, *Zasady muzyki ...*, dz. cyt., s. 39

$\frac{6}{8}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Weak</i> }
$\frac{9}{8}$	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Strong</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Weak</i> }
domyślnie	{ <i>MeasureBeats.Begin</i> , <i>MeasureBeats.Weak</i> , <i>MeasureBeats.Weak</i> , ... }

Tabela 4.2 Rozkład akcentów w takcie w zależności od metrum utworu

#### 4.5.3. Wybór węzłów dla akordów

Dla powstającego zestawienia akordów akompaniamentu należy określić momenty ich zmiany. Podobnie jak z ustalaniem domyślnego rozkładu akcentów, tak i to zadanie jest jedynie bazą dla algorytmu *Harmony Search*, które będzie w stanie ulepszyć pierwotną wersję. Polega ono na nadaniu wag różnym miarom w takcie, a akordy akompaniamentu będą początkowo ustalone dla tych współbrzmień, które posiadają wartość atrybutu *beatStrength* równą *MeasureBeats.Strong* lub *MeasureBeats.Begin*.

Przyporządkowanie wagi rytmicznej danej miary taktu bazuje na nadanych w poprzednim kroku głównych miar. Dodatkowo należy uzupełnić atrybut *beatStrength* dla wszystkich pozostałych akordów brzmienia początkowego, gdyż wpływa on na późniejszy proces dopasowywania akordów. Dźwięki na słabszych miarach w takcie, które nie zostały uwzględnione w początkowym rozkładzie akcentów, nie będą momentami zmiany akordów akompaniamentu, lecz mogą mieć wpływ na jakość tworzonej z nimi harmonii. Nuty o najmniejszych wartościach rytmicznych nie będą brane pod uwagę podczas ustalania akompaniamentu a atrybut *beatStrength* będzie przyjmował wartość *MeasureBeats.Default*. Natomiast te brzmienia, które występują w połowie miar oznaczanych jako silne (*MeasureBeats.Strong*), będą miały przyporządkowany atrybut *beatStrength* o wartości *MeasureBeats.Weak*. Nuty w nich zawarte będą zatem wpływały na dobór akordów akompaniamentu.

#### 4.5.4. Algorytm HS – tworzenie zbioru rozwiązań wejściowych

Algorytm *Harmony Search* został zaadoptowany do rozwiązania problemu bliskiemu swojemu abstrakcyjnemu opisowi, czyli tworzeniu harmonii muzycznej. W tym przypadku, grupie improwizujących na instrumentach muzyków odpowiada zestaw akordów tworzących pełen akompaniament. Zsumowana ocena ich dopasowania do melodii wejściowej jest wartością oceny jakości współbrzmienia muzyków.

Pierwszym krokiem jest stworzenie zestawu rozwiązań początkowych i umieszczenie ich w pamięci harmonii (*Harmony Memory*). Algorytm tworzy nową ścieżkę dźwiękową z akordami akompaniamentu poprzez iteracje po kolekcji współbrzmień otrzymanych z melodii wejściowej (możliwe jest występowanie kilku ścieżek jednocześnie). Akordy akompaniamentu będą

powstawać na miarach taktu oznaczonych jako *MeasureBeats.Begin* lub *MeasureBeats.Strong*, gdyż zbyt częste zmiany podstawy harmoniczej nie są pożądane. Zatem w trakcie trwania pojedynczego akordu akompaniamentu może pojawić się wiele dźwięków melodii głównej. Tak też często są zbudowane utwory muzyczne, iż główna melodia jest bardziej rozbudowana, zawiera krótsze wartości rytmiczne, a akompaniament jest złożony z dłuższych miar. Jest to pewne utrudnienie, gdyż nie wszystkie nuty z zestawu wejściowego są tak samo istotne pod kątem budowy harmonii. Dlatego najkrótsze wartości, które występują na niemiarych częściach taktu swoją rolę rytmiczną mają oznaczoną jako *MeasureBeats.Default* i nie są wówczas brane pod uwagę wyznaczania dopasowania harmonicznego. Wszystkie pozostałe, które występują podczas trwania akordu akompaniamentu wpływają na dobór funkcji harmoniczej proporcjonalnie do długości swojego trwania.

Aby znaleźć pasujące akordy do wskazanego w określonej mierze taktu zestawu brzmień, *MusicAnalyzer* iteruje po wszystkich brzmieniach i ich nutach aby ustalić, które akordy w danym fragmencie nadają się aby tworzyć akompaniament. Funkcja ta sprawdza, czy wszystkie dźwięki brane pod uwagę z melodii głównej zawierają się w kandydującym akordzie lub czy tworzą wraz z nim czterodźwiękowy akord septymowy. Jeśli któryś z dźwięków nie uzupełnia się z akordem w żaden ze wskazanych sposobów, to dany akord jest na tym etapie odrzucany. Tak powstaje lista kandydujących akordów akompaniamentu w danym fragmencie. Na wypadek, gdyby zredukowany w ten sposób zestaw możliwych akordów był pusty, system losowo przydzieli jeden z akordów triady harmoniczej.

Następnie, z listy dostępnych akordów, program wybiera jeden korzystając z losowania metodą ruletki<sup>70</sup>. Każdy z akordów posiada swoją wagę, która określa prawdopodobieństwo wylosowania. Tę informację przechowuje atrybut typu wyliczeniowego *ChordPriority*, który dla odpowiednich funkcji przyjmuje następujące wartości:

Tonika = 6,

Subdominanta = 4,

Dominanta = 4,

Akord na II stopniu = 2,

Akord na VI stopniu = 3,

Akord na III stopniu = 2,

Akord na VII stopniu = 0,

Domyślnie = 0

Dodano także modyfikację w przypadku, gdy w brzmieniu wejściowym występuje tylko jedna nuta. W takim przypadku akord zbudowany na tym dźwięku będzie posiadał najwyższy

---

<sup>70</sup> <http://www.k0pper.republika.pl/geny.htm>, dostęp 20.09.2016

priorytet (równy akordowi toniki). Kolejnym usprawnieniem jest podwojenie prawdopodobieństwa wylosowania odpowiednich akordów na charakterystycznych miarach taktu. Dotyczy to akordu toniki oraz jego akordów zastępczych na początku taktu, a na pozostałych miarach oznaczonych atrybutem *MeasureBeats.Strong* - subdominanty, dominanty oraz ich akordów zastępczych. Faworyzuje to funkcje triady harmoniczej na silnych miarach taktów, gdyż to one i postawione na nich brzmienia mają decydujący wpływ na kształtowaną tonację utworu<sup>71</sup>. Wymienione powyżej akordy zastępcze dla każdego z akordu triady kształtują się następująco:

Akord podstawowy:	Akordy zastępcze <sup>72</sup> :
Tonika	Akord na VI stopniu Akord na III stopniu
Subdominanta	Akord na II stopniu Akord na VI stopniu
Dominanta	Akord na III stopniu (W przypadku tonacji molowej także akord na III stopniu z podwyższonym VII st. gamy)

Tabela 4.3 Akordy zastępcze podstawowych funkcji harmoniczych

Po wybraniu akordu w wyniku selekcji ruletkowej, w określonych przypadkach dodawany jest czwarty dźwięk akordu. Prawdłowo użyty potęguje napięcie budowane w utworze lub tworzy nietypowy dysonans<sup>73</sup>. Z tego względu warto go stosować w momentach prowadzących do kulminacji. W programie *MusicAnalyzer* zastosowano wzbogacenie akompaniamentu w ten sposób, gdyż na ostatniej silnej mierze taktu z 50% prawdopodobieństwem dla wskazanych akordów, dodawany jest 4-ty dźwięk, co tworzy akord septymowy. Dotyczy to następujących akordów:

Moment utworu:	Możliwe akordy septymowe:
Ostatnia miara w takcie w tonacji durowej	Akord dominanty Akord na II stopniu Akord na III stopniu
Ostatnia miara w takcie w tonacji molowej	Akord dominanty Akord na II stopniu

Tabela 4.4 Możliwe akordy septymowe

Kolejnym stosowanym polepszeniem jakości akompaniamentu jest łączenie podobnych akordów. W wielu sytuacjach zmiana akordu na każdej silnej mierze w takcie nie jest konieczna, lecz program początkowo tak czyni. Jeśli sąsiadujące ze sobą akordy są do siebie podobne, to

<sup>71</sup> Franciszek Wesółowski, *Zasady muzyki ...*, dz. cyt., s. 68

<sup>72</sup> K. Sikorski, *Harmonia cz. 2*, Polskie Wydawnictwo Muzyczne, Kraków 1948, s. 114-142

<sup>73</sup> K. Sikorski, *Harmonia cz. 2*, dz. cyt., s. 92

warto je scalić w jeden. Oczywiście nie jest to pożądane w każdym momencie, lecz program nie jest w stanie ich odpowiednio rozróżnić, dlatego wskazana modyfikacja odbywa się z 50% procentowym prawdopodobieństwem. Scalenie dotyczy takich samych akordów bądź akordu podstawowego i jego brzmienia zastępczego oraz ma miejsce w sytuacji, gdy pierwszy z nich jest na silniejszej mierze taktu. Powoduje to wydłużenie mocniej osadzonego brzmienia i zapobiega np. łączeniu akordu z końca taktu z tym w nowym takcie<sup>74</sup>. W wyniku scalenia pozostawiany jest akord o wyższym priorytecie (atrybut typu *ChordPriority*).

Tak powstałe zestawy akordów są zapisywane do obiektu ścieżki dźwiękowej, wyznaczana jest długość ich trwania, uruchamiana jest dla nich funkcja oceny oraz zgodnie z jej wynikami uszeregowane zostają w pamięci harmonicznej (*Harmony Memory*). Powstałe w wyniku opisanych kroków randomizowane rozwiązania odznaczają się dużym zróżnicowaniem układów harmonicznych i co za tym idzie funkcją oceny. Pozwala to wprowadzać liczne optymalizacje w kolejnych krokach algorytmu.

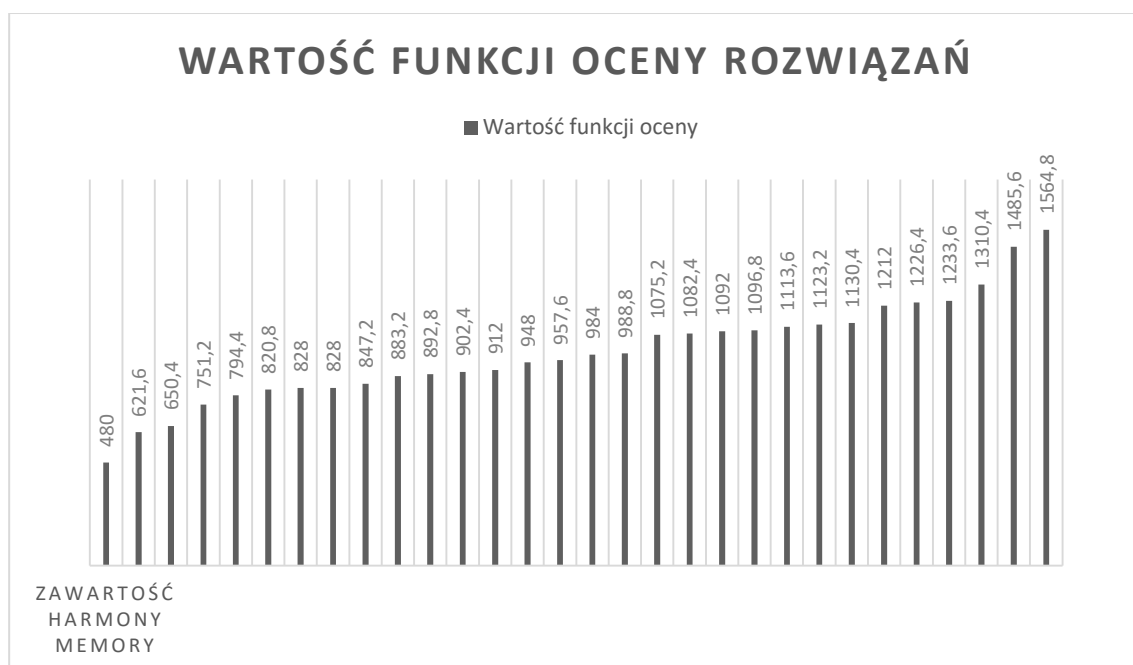


Tabela 4.5 Przykładowe zróżnicowanie rozwiązań początkowych

#### 4.5.5. Algorytm HS – modyfikacje rozwiązań podczas iteracji

Po ustanowieniu podstawowego zestawu rozwiązań, algorytm *Harmony Search* wykonuje iteracje, w których stara się dokonać usprawnień w rozwiązaniach zapisanych w pamięci. W jednej iteracji wybiera jedno rozwiązanie, które będzie starał się usprawnić i następnie zapisać do pamięci, jeśli udało się poprawić najgorszy dotychczasowy wynik. Zgodnie z prawdopodobieństwem określanym przez parametr *HMCR* (ok. 70-80%), wybiera rozwiązanie z

<sup>74</sup> K. Sikorski, *Harmonia* cz. 2, dz. cyt., s. 61

pamięci, lub w przeciwnym razie tworzy nowe, analogicznie jak przy generowaniu rozwiązań początkowych. W tym pierwszym przypadku, dodatkowo z prawdopodobieństwem określonym przez parametr *Pitch Adjusting Rate* (ok. 80-90%), algorytm stara się wykonać określoną ilość losowych zmian, aby w ten sposób poprawić swój rezultat. Ich liczbę określa parametr *Delta* (w zależności od konfiguracji przyjmuje wartości 2 lub 3). Najpierw losowo wskazywany jest akord poddawany zmianie (improwizacji). Następnie do czasu, aż uda mu się zamienić go z dowolnym swoim akordem zastępczym, który posiada inną wagę funkcji harmoniczej, losuje z równym prawdopodobieństwem korzystając z dostępnego zbioru alternatyw.

<b>Akord podstawowy:</b>	<b>Akordy alternatywne<sup>75</sup>:</b>
Tonika	Akord na VI stopniu Akord na III stopniu
Subdominanta	Akord na II stopniu Akord na VI stopniu
Dominanta	Akord na III stopniu (W przypadku tonacji molowej także akord na III stopniu z podwyższonym VII st. gamy)
Akord na VI st. gamy	Tonika Subdominanta
Akord na II st. gamy	Subdominanta
Akord na III st. gamy	Dominanta Tonika (W przypadku tonacji molowej także akord na III stopniu z podwyższonym VII st. gamy)

*Tabela 4.6 Zestaw akordów alternatywnych dla procesu improwizacji*

W przypadku, gdy wybrane w danej iteracji rozwiązanie powstało od nowa, opisana powyżej improwizacja (losowa zmiana akordów) nie jest wykonywana. Ustalone w jeden z powyższych sposobów nowe brzmienie akompaniamentu zostaje poddane ponownej ocenie przez funkcję obliczającą sumę karnych punktów. Następnie, jeśli jest lepsze od najgorszego dotąd w pamięci harmonii, wówczas zastępuje je a cały zestaw zostaje poddany sortowaniu. Progres najlepszego wyniku w pamięci jest największy na początku działania algorytmu, natomiast losowe modyfikacje w późniejszych iteracjach zdarzają się bardzo nieregularnie, przez co konieczne jest zastosowanie bardzo wielu iteracji. Ich przykładowy przebieg zaprezentowany jest na poniższym diagramie.

<sup>75</sup> K. Sikorski, *Harmonia cz. 2*, dz. cyt., s. 114-142



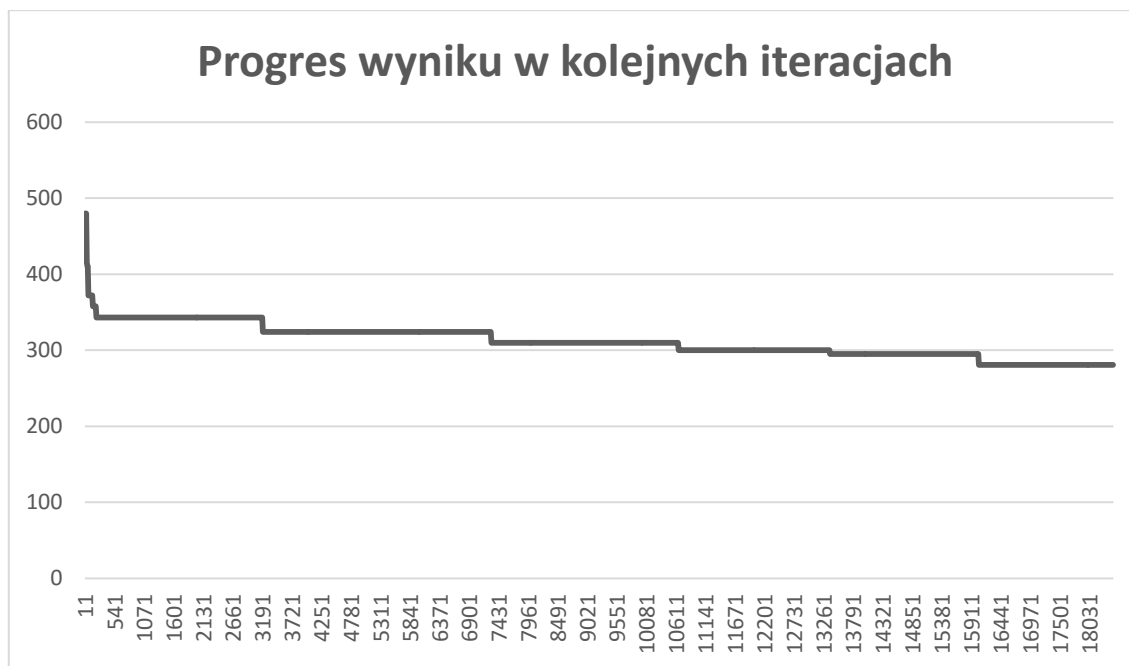


Tabela 4.7 Progres wyniku najlepszego rozwiązania w iteracjach algorytmu

#### 4.5.6. Algorytm HS – funkcja oceny i warunki końcowe

Do oceny jakości rozwiązania program wykorzystuje funkcję sprawdzającą wiele atrybutów. Przede wszystkim obliczana jest kara za niezgodne współbrzmienia z melodią wejściową. Po odrzuceniu dźwięków oznaczonych jako nieistotne ze względu na budowaną harmonię, wszystkie jednocześnie brzmienia akompaniamentu i melodii głównej są oceniane pod kątem jakości współbrzmienia wedle poniższych kryteriów:

- Jeśli akompaniament zawiera wszystkie dźwięki z melodii głównej, nie dostaje karnych punktów. W przeciwnym wypadku, sprawdzany jest typ akordu, który tworzy złożenie brzmienia melodii głównej z akompaniamentem. Możliwe wartości to:
  - Trójdźwięk Durowy
  - Trójdźwięk Molowy
  - Trójdźwięk Zmniejszony (dwie tercje małe)
  - Trójdźwięk Zwiększony (dwie tercje wielkie)
  - Dominanta septymowa (trójdźwięk durowy bądź molowy z dodaną tercją)
  - Zmniejszony septymowy (trzy tercje małe)
  - Inny (pozostałe)

Jeśli otrzymujemy nieznane współbrzmienie (typ *Inny*), to otrzymuje ono 2 pkt kary. W pozostałych przypadkach kara nie jest naliczana.

- Faworyzowane są dłuższe brzmienia akordów triady harmoniczej<sup>76</sup>. Jeśli akord trwa dłużej niż średnia, a nie jest jednym z powyższych, otrzymuje karny 1 pkt.
- Lepiej oceniane są krótsze brzmienia akordów pochodnych. Jeśli akord trwa krócej niż średnia, a jest akordem z triady harmoniczej, otrzymuje karny 1 pkt.
- Jeśli poprzedni akord był na silniejszej części taktu, a miał mniej istotną funkcję harmoniczną określoną przez typ wyliczeniowy *ChordPriority*, to wówczas nadawana jest kara 1 pkt.
- Analogicznie, gdy ostatni akord miał wyższy priorytet, a był na słabszej mierze taktu, też jest nadawana kara 1 pkt.
- Jeśli poprzedni i obecny akord są tożsame, to może być to zaletą i wadą jednocześnie. Jeśli przedłużono wartość ze słabej miary taktu, jest nadawana kara 1 pkt, a gdy poprzedni akord był na silniejszej części taktu i został powielony, to następuje gratyfikacja 1 pkt.
- Jeśli na ostatniej mierze taktu występuje akord, który nie jest jednym z możliwych czterodźwiękowych akordów septymowych, to nadawana jest kara 1 pkt.
- Jeśli czterodźwiękowy akord septymowy występuje na innej części taktu niż ostatnia miara, to nadawana jest kara 1 pkt.
- Jeśli pojawiłby się pusty akord, to nadawana jest kara 100 pkt, jeśli miałby 2 dźwięki, to dostaje karę 5pkt, a gdy tylko 1, to 10 pkt kary.

Wyliczona kara za niezgodność obu brzmień jest następnie przemnażana przez długość swojego trwania i dodawana do całościowej oceny. Ten krok jest powtarzany dla wszystkich współbrzmień melodii wejściowej z akordami akompaniamentu i pozwala obliczyć całkowitą karę. Najlepsze rozwiązanie posiada najmniejszą sumę karnych punktów.

Nie jest możliwe określenie optymalnego dopasowania akordów, lecz ciąg iteracji algorytmu *HS* polepsza jakość rozwiązań zawartych w pamięci harmonii względem określonych kryteriów. Ten proces po pewnym czasie może jednak nie przynosić rezultatów a stosowanie losowych modyfikacji nie polepsza najlepszego rezultatu. Dla takiego przypadku warto ustanowić próg graniczny nieudanych liczby iteracji. W programie *MusicAnalyzer* odpowiada za to parametr *bestTrackUnchanged*, który przy testach jakościowych przyjmował wartość 20000 iteracji. Kolejnym warunkiem granicznym, który obejmuje algorytm, jest czas wykonywania obliczeń. Przy testach jakościowych był ustawiany bardzo duży limit (500 sekund), natomiast przy testach praktycznych było to główne ograniczenie algorytmu i wynosiło 3 sekundy. Jest to wartość, która przy standardowym użytkowaniu jest wygodna dla użytkownika. Ostatnim warunkiem stopu pętli algorytmu było osiągnięcie maksymalnej zakładanej liczby iteracji. Testy programu wskazywały, iż nieznaczne poprawy wyniku są możliwe nawet po 50000 iteracji, zatem granicę końcową ustawiono na 80000, a dla testów praktycznych ustawień jednie 1000. Główna pętla algorytmu

---

<sup>76</sup> K. Sikorski, *Harmonia cz. 2*, dz. cyt., s. 76

wykonującą improwizację i tworzącą nowe randomizowane rozwiązania kończy się po osiągnięciu dowolnego z trzech wymienionych kryteriów stopu.

#### **4.5.7. Ustalenie atrybutów nut akompaniamentu**

Po wskazaniu funkcji harmonicznego akompaniamentu, należy przypisać im konkretne wartości nut w odpowiedniej oktawie. Mimo, że funkcje harmoniczne można odwzorować w każdej oktawie, jej wybór nie jest bez znaczenia. Często regułą np. w dziełach orkiestrowych jest to, iż melodie prowadzące grane są przez instrumenty operujące na wysokich rejestrach, a te drugorzędne są zapisane dla rejestrów niższych<sup>77</sup>. Stąd najczęściej główną frazę w orkiestrze prowadzi skrzypce, czasami przejmują flety, a za podkład oraz bazę rytmiczną i harmoniczną odpowiedzialne są wiolonczele i kontrabasy. Nie jest to bynajmniej faworyzowanie niektórych instrumentów, lecz ma to swoje podstawy w tym, jak ludzkie ucho odbiera dźwięki. Te o wyższych częstotliwościach są lepiej przez nas słyszane<sup>78</sup>, są głośniejsze i lepiej rozchodzą się po mniejszej przestrzeni. Dźwięki niższe natomiast są cichsze, a przez niewielkie różnice częstotliwości między bliskimi sobie dźwiękami są dużo gorzej rozróżnialne przez zmysł słuchu człowieka<sup>79</sup>.

Program *MusicAnalyzer* dostosowuje się także do naturalnych zdolności słuchowych człowieka, dlatego tworzona kompozycja najczęściej nie pokrywa się z melodią główną, lecz występuje w niższej oktawie. Ze względu na charakter testowy aplikacji, która przed wszystkim ma służyć do uruchomienia procesu kompozycji, odsłuchania i ocenienia końcowego efektu w postaci utworu *MIDI*, nie mogą to jednak być dźwięki zbyt niskie. Wówczas ocena rozwiązania byłaby trudniejsza, gdyż niższe dźwięki, których źródłem są proste próbki instrumentów elektronicznych zlewałyby się w jedno wspólne brzmienie. Zastosowano zatem średni rejestr: najniższym możliwym dźwiękiem akompaniamentu jest F3, natomiast najwyższym D5. Tak zbudowane akordy nie przykrywają brzmienia melodii głównej a jednocześnie są wyraźne i pozwalają łatwo rozpoznać harmonię, którą wspólnie tworzą.

Drugim etapem procesu ustalania atrybutów nut powstałego akompaniamentu jest dobór przewrotu każdego z akordów. Nie ma to kluczowego znaczenia w zakresie harmonii, natomiast wpływa na estetykę, płynność a także na trudność wykonania kompozycji. Bywają sytuacje, w których kompozytorzy celowo rozpraszają akordy między oktavami, co może potęgować wrażenie dramatyzmu bądź powagi w danym fragmencie<sup>80</sup>. Przykładem takiego zabiegu jest poniższy zapis nutowy prezentujący III Koncert Fortepianowy S. Rachmaninowa.

---

<sup>77</sup> Michał Zieliński, *Technika orkiestrowa w utworach Karola Szymanowskiego*, Wydawnictwo uczelniane AM., Bydgoszcz 1993, s. 35

<sup>78</sup> Elżbieta Sławińska, *Przetwarzanie dźwięku przez ucho*, Państwowa Wyższa Szkoła Muzyczna, Warszawa 1968, s. 7

<sup>79</sup> Elżbieta Sławińska, *Przetwarzanie dźwięku...*, dz. cyt., s. 7

<sup>80</sup> Barbara Smoleńska-Zielińska, *Przeżycie estetyczne muzyki*, WSiP, Warszawa 1991, s. 56



Rysunek 4.17 Fragment III Koncertu Fortepianowego S. Rachmaninowa, źródło:  
[https://upload.wikimedia.org/wikipedia/commons/thumb/d/db/Piano\\_Concerto\\_3\\_Cadenza.png/350px-Piano\\_Concerto\\_3\\_Cadenza.png](https://upload.wikimedia.org/wikipedia/commons/thumb/d/db/Piano_Concerto_3_Cadenza.png/350px-Piano_Concerto_3_Cadenza.png)

*MusicAnalyzer* w swych założeniach bazuje jednak na prostocie prezentacji zapisu muzycznego, aby powstający akompaniament był czytelny i łatwy do oceny. Posiada zatem rozwiązania wspomagające zarówno płynność jak i łatwość wykonania utworu. Można je uzyskać w bardzo podobny sposób, gdy dźwięki kolejnych akordów są do siebie bardzo zbliżone, a w szczególności gdy niektóre z nich się powtarzają. Takie ułożenie akordów powoduje wrażenie u słuchacza, iż są one częścią wspólnej melodii i zachowują jej cechy, ponieważ nie różnią się diametralnie od siebie<sup>81</sup>. Program *MusicAnalyzer* szuka zatem najlepiej pasującego przewrotu dla akordu tak, aby różnił się on od poprzedniego w możliwie najmniejszy sposób. Tę ocenę uzyskuje poprzez porównanie dźwięków akordu poprzedzającego z bieżącym w różnych przewrotach. Sumowane są odległości między odpowiadającymi sobie dźwiękami i wybierany jest akord w tym przewrocie, dla którego suma odległości jest najmniejsza.

Pozostałe cechy nut potrzebne do ich prawidłowej prezentacji, ustalane są w sposób analogiczny jak dla ścieżki dźwiękowej zawartej w pliku wejściowym.

#### 4.6. Zapis nutowy i dźwiękowy powstałego akompaniamentu

Po zakończeniu działania algorytmu *Harmony Search*, *MusicAnalyzer* wybiera najlepsze utworzone rozwiązanie i prezentuje jego zapis nutowy jako kolejną ścieżkę dźwiękową widoczną na ekranie głównym aplikacji. Zawarte w niej nuty posiadają analogicznie te same atrybuty, co zdekodowane dźwięki ze ścieżek z pliku wejściowego.

Możliwe jest także wydrukowanie zapisu nutowego akompaniamentu w formacie pionowym i w podziale na strony w zależności od długości zapisu. Opcja ta dostępna jest z menu głównego ekranu (*Menu -> Print accompaniment*). Należało w tym przypadku dokonać pewnych przekształceń, gdyż element graficzny w oknie aplikacji prezentuje nuty w jednym wierszu, co w przypadku prezentacji na wydruku nie byłoby funkcjonalne. W tym celu należało podzielić jedną pięciolinie na wiele systemów tak, aby w pełnej wielkości mogły się one pomieścić na kartce papieru. Aby podzielić element graficzny na wiele części należało wpierw zapisać jego zawartość

<sup>81</sup> Barbara Smoleńska-Zielińska, *Przeżycie estetyczne...*, dz. cyt., s. 86

jako bitmapę a następnie kolejne fragmenty skopiować do nowej bitmapy, której wymiary umożliwiają pomieszczenie wielu systemów z pięcioliniami. Dodatkowo należało zastosować podział na strony w przypadku, gdy zapis miałby zajmować więcej niż 1 stronę. Pomocny okazał się artykuł<sup>82</sup> Benjamina Walkera, w którym opisuje on sposób implementacji podziału wydruku na strony przy konwersji elementów graficznych na bitmapy.

Przy użyciu narzędzi systemowych *Windows*, wydruk można zapisać do pliku *.pdf*. Przykładowy wydruk jest przedstawiony na poniższej ilustracji. Brak bocznych marginesów jest zdeterminowany tym, iż wskazane narzędzie zapisujące plik w formacie *.pdf* umożliwia wydruk na całej stronie, bez marginesów. Przy użyciu fizycznej drukarki, ilość wierszy i ich szerokość będzie dostosowana do możliwości drukarki.



*Rysunek 4.18 Przykładowy wydruk zapisu nutowego akompaniamentu*

Program pozwala także zapisać powstałą ścieżkę dźwiękową do pliku *MIDI*. Zostaje ona dodana do ścieżek istniejących i w takiej formie możemy nadpisać plik wejściowy lub utworzyć nowy. Program umożliwia wówczas także odtworzenie melodii bazowej wraz z akompaniamentem, co ułatwia ocenę jego dopasowania.

---

<sup>82</sup> <http://www.codeproject.com/Articles/339416/Printing-large-WPF-UserControls>, dostęp 08.09.2016

Zapisanie nowej ścieżki do pliku *MIDI* jest wspierane przez bibliotekę *Sanford.Multimedia*<sup>83</sup> poprzez dodanie obiektu klasy *Track* do obiektu *Sequence*, który jest kolekcją ścieżek dźwiękowych. Wpierw należy stworzyć ścieżkę z odpowiednimi dźwiękami oraz z oznaczeniem wyboru instrumentu, którego próbki mają być wykorzystane do odtworzenia melodii. W tym celu należy dodać na samym początku ścieżki obiekt *ChannelMessage*, dla którego typ komendy to *ProgramChange* (jest to oznaczenie zmiany instrumentu zgodne ze standardem *MIDI*). Obiekt ten należy stworzyć przy pomocy obiektu *ChannelMessageBuilder*, aby utworzyć odpowiedni ciąg bajtów do zapisu w pliku binarnym. Jednym z argumentów jest numer instrumentu z zakresu <0; 127> zgodny z podstawowym standardem *MIDI*<sup>84</sup> (np. *AcousticGrandPiano* = 0). Następnie należy dodać wszystkie nuty należące do wskazanej ścieżki. Muszą być umieszczone we wskazanym miejscu w czasie oraz dla każdej z nich należy nadać komendy *NoteOn* (włączenie dźwięku) i *NoteOff* (wyłączenie). Do ich wygenerowania program korzysta tak samo jak we wcześniejszym przypadku z obiektu *ChannelMessageBuilder*.

#### 4.7. Stosowane rozwiązania pomocnicze

W aplikacji *MusicAnalyzer* stosowane były rozwiązania udostępniane przez ogólnodostępne biblioteki działające zgodnie z licencją *Open Source*, dzięki czemu mogą być używane do celów niekomercyjnych. Dołączenie ich referencji jest konieczne do skompilowania projektu.

##### 4.7.1. Sanford.Multimedia

*Sanford.Multimedia*<sup>85</sup> to biblioteka dla platformy .Net udostępniająca klasy i metody ułatwiające odczytywanie i analizę plików dźwiękowych w formacie *Musical Instrument Digital Interface (MIDI)*. Projekt autorstwa Leslie Sanford umożliwia wczytanie pliku wejściowego, odkodowanie zawartych w nim sygnałów oraz odtworzenie zapisanej w nim muzyki. Poniżej przedstawione zostały wybrane klasy, które były używane w projekcie *MusicAnalyzer* i dostarczały funkcjonalności potrzebnych do operacji na pliku *MIDI*:

- **MidiEvent** – odzwierciedla zdarzenia (sygnały) zachodzące w określonych momentach czasu, które zawierają różnorakie informacje związane z daną chwilą; przykładem takich zdarzeń są przykładowo: zmiana tonacji, zmiana tempa, początek trwania określonego dźwięku lub jego koniec;
- **Track** – odpowiada pojedynczej ścieżce dźwiękowej zapisanej w pliku; zawiera w sobie kolekcję obiektów klasy **MidiEvent**, czyli zdarzeń zaistniałych na danej ścieżce;

---

<sup>83</sup> Zob. Punkt 4.7.1 niniejszej pracy

<sup>84</sup> <https://www.midi.org/specifications/item/gm-level-1-sound-set>, (dostęp 14.09.2016)

<sup>85</sup> L. Sanford, *C# MIDI Toolkit*, 18.04.2007, <http://www.codeproject.com/Articles/6228/C-MIDI-Toolkit>, dostęp 14.09.2016

- **Sequence** – odpowiada całości zawartości pliku MIDI i posiada kolekcję obiektów typu **Track**, przez co umożliwia zapisywanie, odczytywanie i odtwarzanie pliku muzycznego w całości;
- **ChannelMessage** – wiadomość zawarta w zdarzeniu **MidiEvent**, która jest przypisana do konkretnego kanału dźwiękowego (stereo); są to głównie wiadomości o rozpoczęciu (*NoteOn*) lub zakończeniu (*NoteOff*) dźwięku o wskazanej wysokości i głośności;
- **MetaMessage** – wiadomości pozostałych typów, które nie są bezpośrednio związane z danym kanałem, lecz ze ścieżką dźwiękową (np. zmiana tonacji)
- **OutputDevice** – klasa opisująca urządzenie umożliwiające wysyłanie sygnałów MIDI, aby móc odsłuchać utwór muzyczny
- **Sequencer** – umożliwia odtwarzanie sygnałów muzycznych zapisanych w obiekcie **Sequence**; tworzy kolejkę sygnałów i wysyła je poprzez **OutputDevice**

#### 4.7.2. PSAMControlLibrary

Jest to biblioteka<sup>86</sup> funkcji umożliwiających prezentację graficzną zapisu nutowego w technologii *Windows Forms* na platformie *.Net*. Projekt autorstwa Jacka Salamona posiada też bardziej rozbudowaną wersję komercyjną, lecz używana w tym projekcie biblioteka jest dostępna w ramach licencji *BSD*. Udostępnia ona obiekty odwzorowujące elementy zapisu melodii na pięciolinii oraz umożliwia wygenerowanie widoku zapisu nutowego w formacie w technologii *Windows Forms*. Kilka używanych obiektów oraz typów wyliczeniowych:

- **Clef** – klucz na pięciolinii (basowy lub wiolinowy)
- **Key** – zestaw znaków przy kluczu odpowiadający tonacji utworu (krzyżyki lub bemole)
- **TimeSignature** – oznaczenie metrum (np.  $\frac{3}{4}$ )
- **Note** – nuta na pięciolinii o odpowiedniej wysokości i wartości rytmicznej
- **MusicSymbolDuration** – wartość rytmiczna nuty (np. ósemka, ćwierćnuta)
- **BarLine** – kreska taktowa
- **Rest** – pauza odpowiedniej długości (np. pauza ósemkowa, ćwierćnutowa)

#### 4.7.3. PSAMWPFCControlLibrary

Opisywana powyżej biblioteka została stworzona z myślą o interfejsie graficznym w technologii *Windows Forms*. Jako że *MusicAnalyzer* tworzy okno aplikacji dzięki *Windows*

---

<sup>86</sup> Jacek Salamon, *PSAM Control Library*, 24.06.2010,  
<http://www.codeproject.com/Articles/87329/PSAM-Control-Library>, dostęp 22.03.2016 r.

*Presentation Foundation*, konieczne było wykorzystanie biblioteki *PSAMWPFControlLibrary*<sup>87</sup>, która jest przystosowana do budowy interfejsu w tej technologii i daje też dużo szersze możliwości graficzne. Dzięki dobrej kompatybilności *Windows Forms* i *WPF*, można było użyć elementów muzycznych z pierwotnej biblioteki, natomiast konieczne było jedynie osadzenie ich w elemencie typu *IncipitViewerWPF*, który gromadził na pięciolinii dodawane kolejno do niego elementy.

Aby wymienione powyżej znaki pokazywały się prawidłowo, należy w systemie operacyjnym zainstalować czcionkę *Polihymnia* dołączonej do projektu i dystrybuowanej zgodnie z licencją *Sil Open Font Licence*.

---

<sup>87</sup> Jacek Salamon, *PSAM WPF Control Library*, 24.06.2010, <http://www.codeproject.com/Articles/89582/PSAM-WPF-Control-Library>, dostęp 22.03.2016



## 5. Ocena

### 5.1. Założenia testów

Sposób oceny tworzenia akompaniamentu przez program *MusicAnalyzer*:

- Jako dane źródłowe zostały użyte 6 różnych melodii jedno- bądź dwugłosowych:
  - Oberek powiślański (dwugłos: sopran i bas) – *OberekPowiślańskiSB.mid*
  - Hymn Włoch (melodie dwóch instrumentów) - *National Anthem - Italy Melody.mid*
  - Hymn Stanów Zjednoczonych (melodie dwóch instrumentów) - *National Anthem - USA Melody.mid*
  - Główny motyw muzyczny z filmu *Piraci z Karaibów* - *Pirates of the Caribbean Melody.mid*
  - Hymn Francji (główna melodia) - *National Anthem – France Melody.mid*
  - Hymn Wielkiej Brytanii (główna melodia) - *National Anthem - UK Melody.mid*

Wszystkie wejściowe i wyjściowe pliki dźwiękowe są dostępne w folderze *Midis* dostępnym w głównym folderze projektu *MusicAnalyzer*.

- Dla każdego testu zostały wywołane trzy uruchomienia programu, a ocenie zostało poddane jedynie to dające najlepsze efekty, co zostało określone na podstawie funkcji oceny w algorytmie harmonicznym. Uczyniono tak, gdyż *Harmony Search* to algorytm dający niedeterministyczne rozwiązania, dzięki czemu przy każdym jego uruchomieniu nawet przy tych samych parametrach, wyniki mogą nieznacznie różnić się od siebie.
- Wszystkie testy programu będą wykonywane na tym samym komputerze, bez obciążenia pracą innych aplikacji, przy zastosowaniu dwóch ustawień parametrów:
  - I wariant – test jakościowy (przy wykorzystaniu wystarczająco dużych zasobów)
    - *HMS (Harmony Memory Size) = 50*
    - *HMCR (Harmony Memory Consideration Rate) = 0,8*
    - *PAR (Pitch Adjusting Rate) = 0,7*
    - *Delta = 2*
    - *MaxIter (Maximum number of iterations) = 80000*
    - *BestTrackUnchanged (Maximum number of iterations without best value change) = 20000*
    - *MaxTime = 500(s)*

Dobranie powyższych wartości zostało utworzone na podstawie badań empirycznych, przy jakich ustawieniach program jest w stanie stworzyć najbardziej wydajną kompozycję.

- II wariant – test jakości przy zachowaniu wydajności działania; głównym kryterium ograniczającym będzie czas działania algorytmu = 3 s. Rzutuje ono

także na wielkość zbioru rozwiązań, gdyż nie może być ich zbyt wiele, by iteracje wykonywały się szybko.

- *HMS (Harmony Memory Size)* = 20
- *HMCR (Harmony Memory Consideration Rate)* = 0,5
- *PAR (Pitch Adjusting Rate)* = 0,9
- *Delta* = 3
- *MaxIter (Maximum number of iterations)* = 1000
- *BestTrackUnchanged (Maximum number of iterations without best value change)* = 1000
- *MaxTime* = 3(s)

Dobór powyższych wartości został utworzony na podstawie badań empirycznych, przy jakich ustawieniach program działa w przystępnym czasie dla użytkownika przy zachowaniu maksymalnej możliwej jakości. Ważne jest zatem osiągnięcie zadowalającego rezultatu w bardzo krótkim czasie, a więc po małej ilości iteracji. Dlatego zwiększono ilość powstających nowych rozwiązań oraz ich zmiany w wyniku improwizacji (*HMCR*, *PAR* \* *Delta*).

- Do oceny działania każdej wygenerowanej kompozycji zostało powołanych 5 osób na co dzień związanych z wykonywaniem muzyki klasycznej, aby oceniły działanie programu wedle ustalonych kryteriów. Nie wszystkie osoby potrafią czytać zapis nutowy. Uśrednienie wyników ich oceny umożliwi wskazanie względnie obiektywnej oceny działania programu w różnych aspektach i pod różnymi ustawieniami.

## 5.2. Kryteria oceny

Osoba testująca ocenia każdą kompozycję nadając punkty <1; 6> pod kątem następujących kryteriów:

- Harmoniczne współbrzmienie akompaniamentu z melodią główną (1 – bardzo słabe, 6 – perfekcyjne)
- Rytm zmieniania się akordów (1 – rytm akompaniamentu zupełnie nie odpowiada rytmowi melodii bazowej, 6 – perfekcyjne dopasowanie momentów zmian akordów)
- Podobieństwo akompaniamentu do melodii wejściowej (1 – akompaniament jest uderzająco podobny do melodii bazowej, 6 – akompaniament sprawia wrażenie osobnej lecz dobrze współbrzmiejącej melodii)
- Różnorodność współbrzmień (1 – powtarzalne akordy tworzone na tonice dźwięku melodii bazowej, 6 – ciekawe, innowacyjne współbrzmienia, które też nie wykraczają poza ramy harmonii klasycznej)

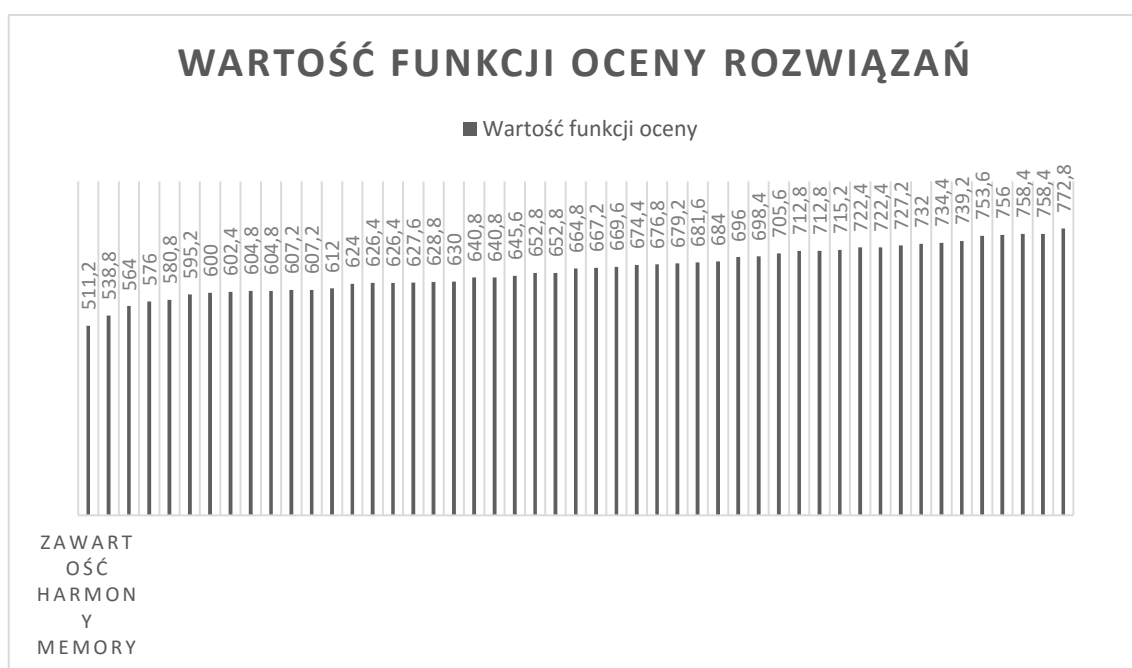
- Ogólna ocena współbrzmienia akompaniamentu z melodią główną (1 – akompaniament zupełnie nie pasuje i jest nieciekawym, sztuczny, 6 – świetnie dopasowany akompaniament naśladowujący twórczość rzeczywistego kompozytora)

### 5.3. Wyniki testów

#### 5.3.1.1 wariant – testy jakościowe

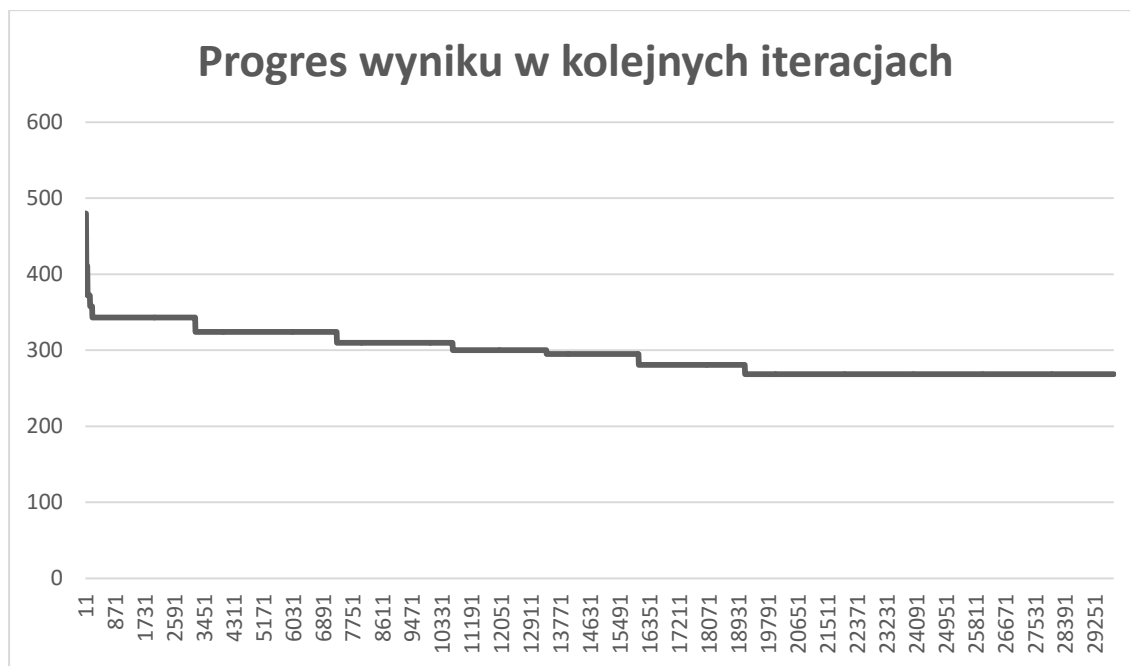
#### 1. Oberek powiślański (dwugłos: sopran i bas):

- Zróżnicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.1 Oberek Powiślański - zróżnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.2 Oberek powiślański - progres najlepszego wyniku w iteracjach

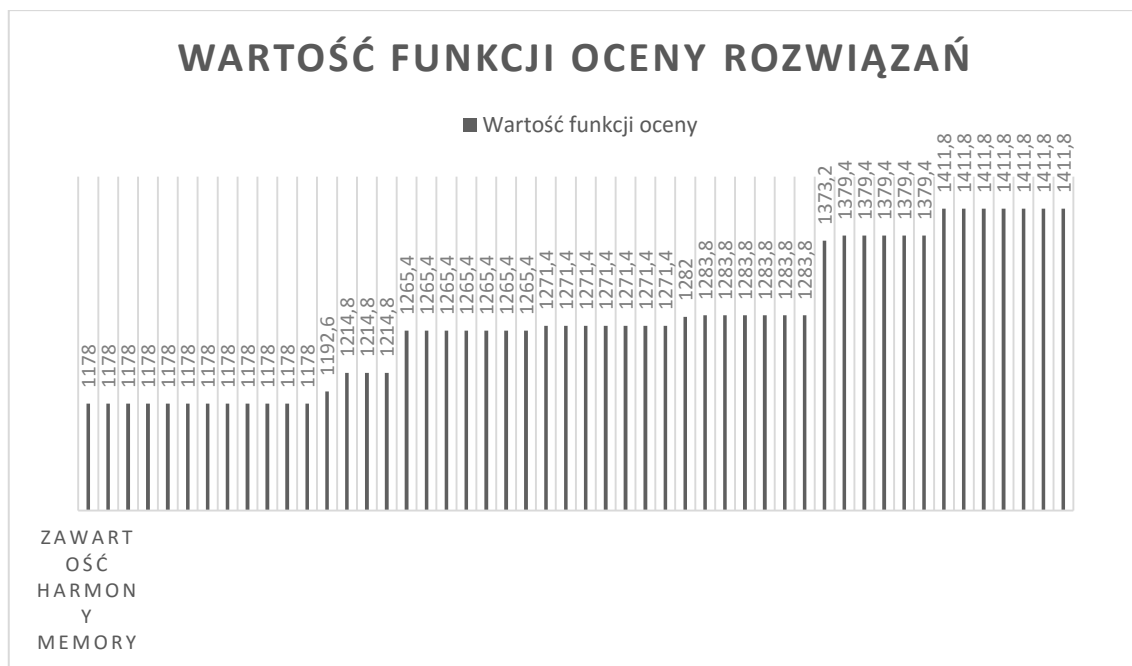
- Ocena wybranego rozwiązania:

	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
Harmoniczne współbrzmienie akompaniamentu z melodią główną	5	4	5	4	4
Rytm zmieniania się akordów	3	3	4	2	3
Podobieństwo akompaniamentu do melodii wejściowej	3	3	4	2	3
Różnorodność współbrzmień	4	3	4	4	3
<b>Ogólna ocena</b>	<b>4</b>	<b>3</b>	<b>4</b>	<b>3</b>	<b>3</b>

Tabela 5.1 Ocena testu wydajnościowego dla Oberka Powiślańskiego

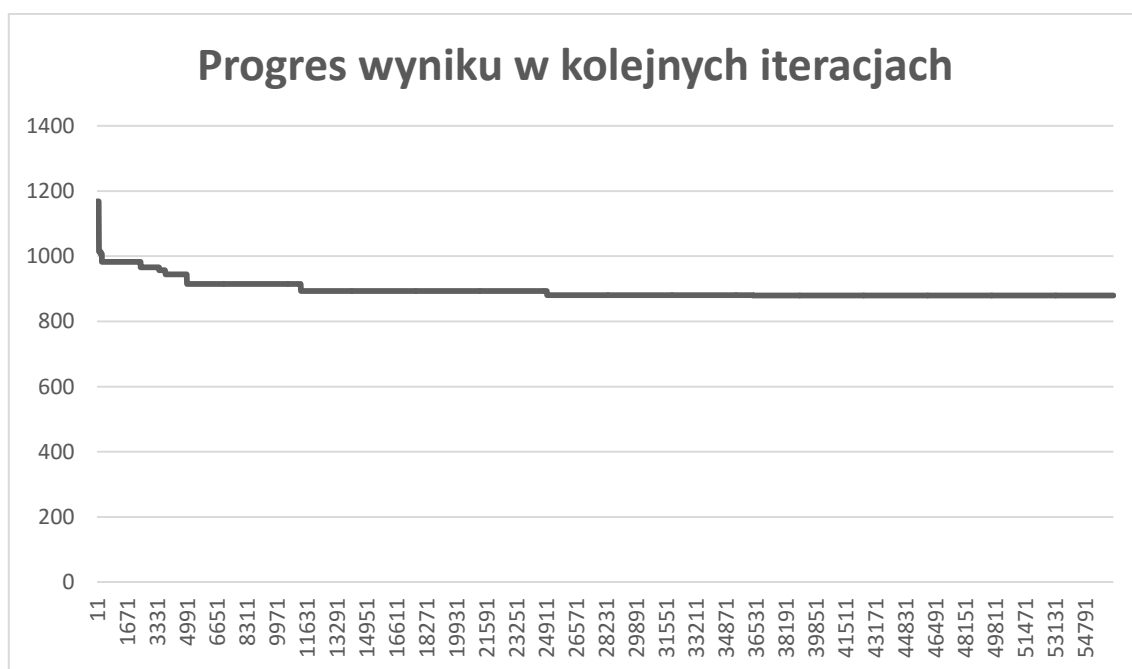
## 2. Hymn Włoch (melodie dwóch instrumentów):

- Zróżnicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.3 Hymn Włoch - zróżnicowanie rozwiązań pocztkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:

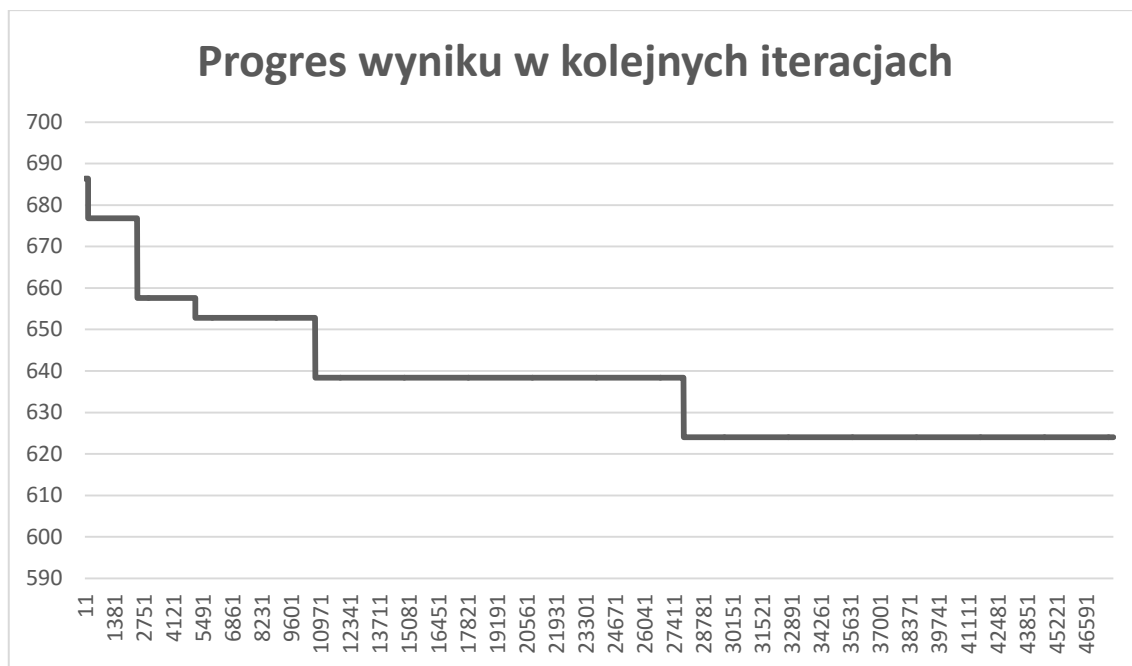


Rysunek 5.4 Hymn Włoch - progres najlepszego wyniku w iteracjach

- Ocena wybranego rozwiązania:

	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
--	---------	---------	---------	---------	---------





*Rysunek 5.6 Hymn USA - progres najlepszego wyniku w iteracjach*

- Ocena wybranego rozwiązania:

	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
Harmoniczne współbrzmienie akompaniamentu z melodią główną	5	4	5	5	4
Rytm zmieniania się akordów	4	3	3	3	3
Podobieństwo akompaniamentu do melodii wejściowej	4	3	4	4	3
Różnorodność współbrzmień	5	4	4	5	4
<b>Ogólna ocena</b>	<b>5</b>	<b>4</b>	<b>4</b>	<b>5</b>	<b>4</b>

*Tabela 5.3 Ocena testu wydajnościowego dla Hymnu USA*

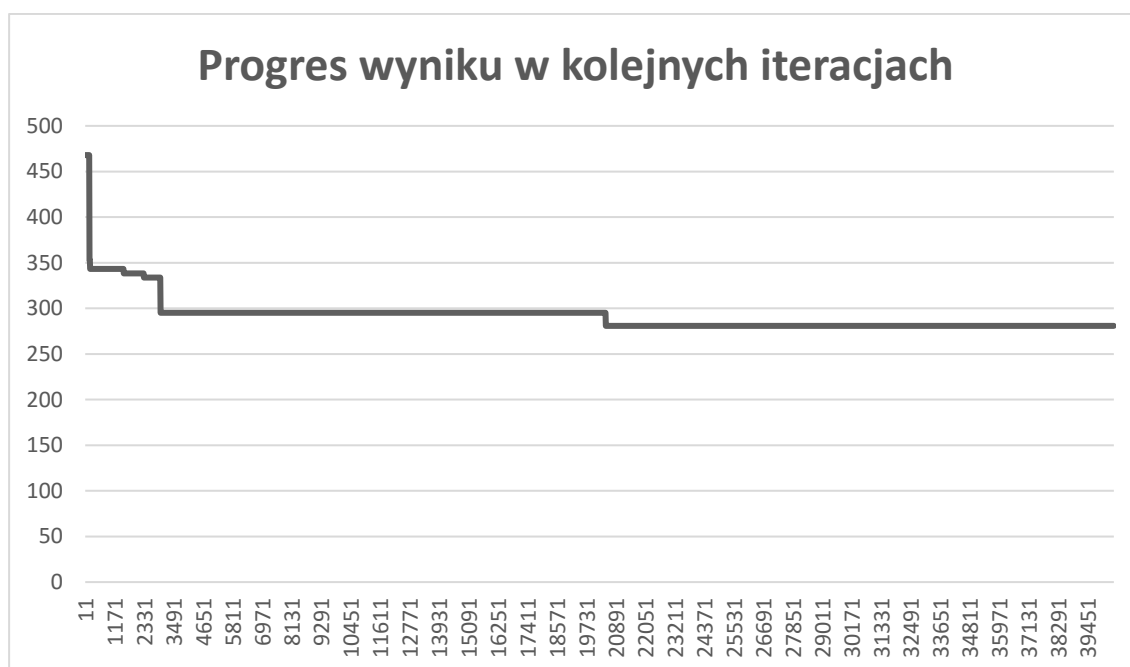
#### 4. Główny motyw muzyczny z filmu „Piraci z Karaibów”

- Zróżnicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.7 Piraci z Karaibów - zróżnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.8 Piraci z Karaibów - progres najlepszego wyniku w iteracjach

- Ocena wybranego rozwiązania:

	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
--	---------	---------	---------	---------	---------

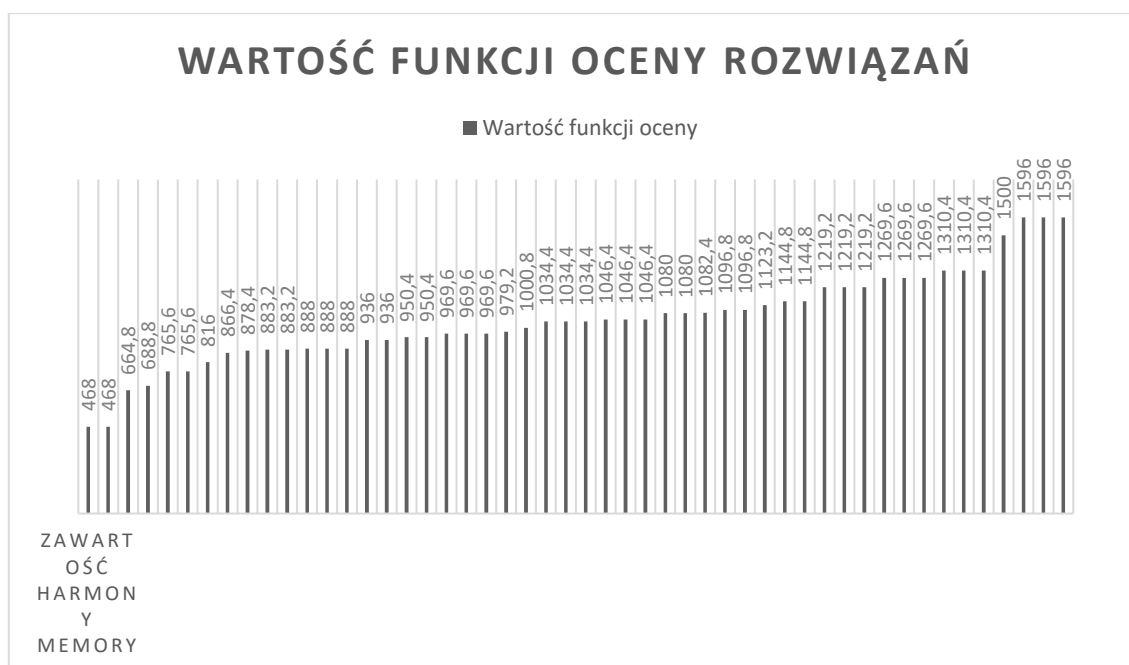


Harmoniczne współbrzmienie akompaniamentu z melodią główną	5	5	6	6	5
Rytm zmieniania się akordów	6	5	5	6	5
Podobieństwo akompaniamentu do melodii wejściowej	5	4	5	5	5
Różnorodność współbrzmień	5	4	5	6	5
<b>Ogólna ocena</b>	<b>5</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>5</b>

Tabela 5.4 Ocena testu wydajnościowego dla Piratów z Karaibów

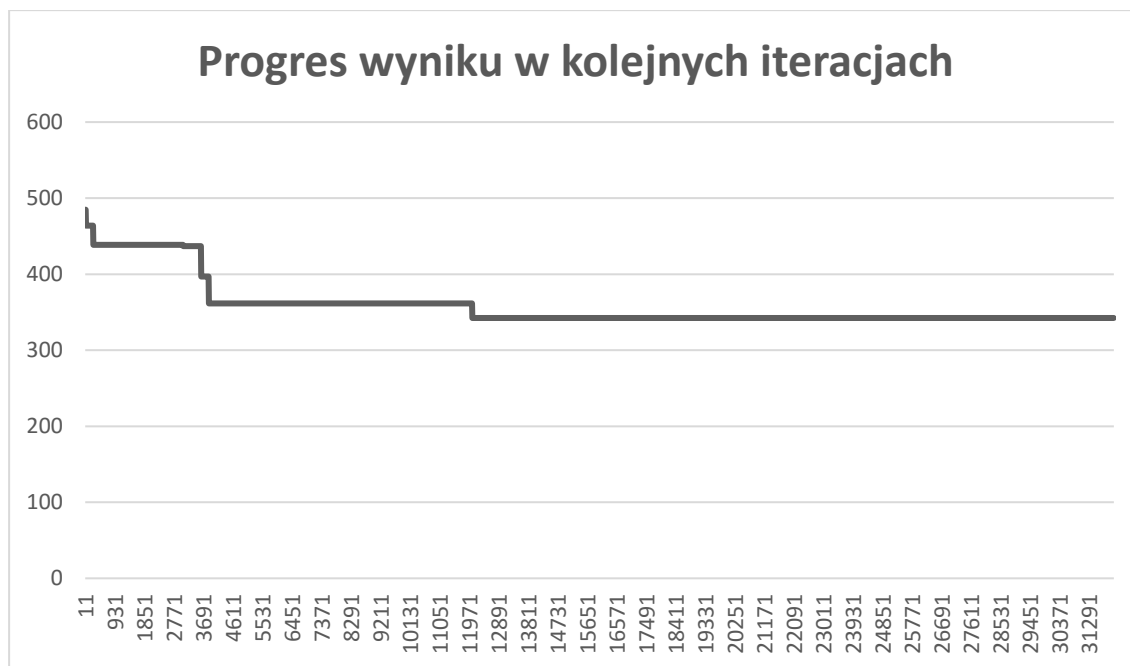
## 5. Hymn Francji (główna melodia)

- Zróżnicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.9 Hymn Francji - zróżnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.10 Hymn Francji - progres najlepszego wyniku w iteracjach

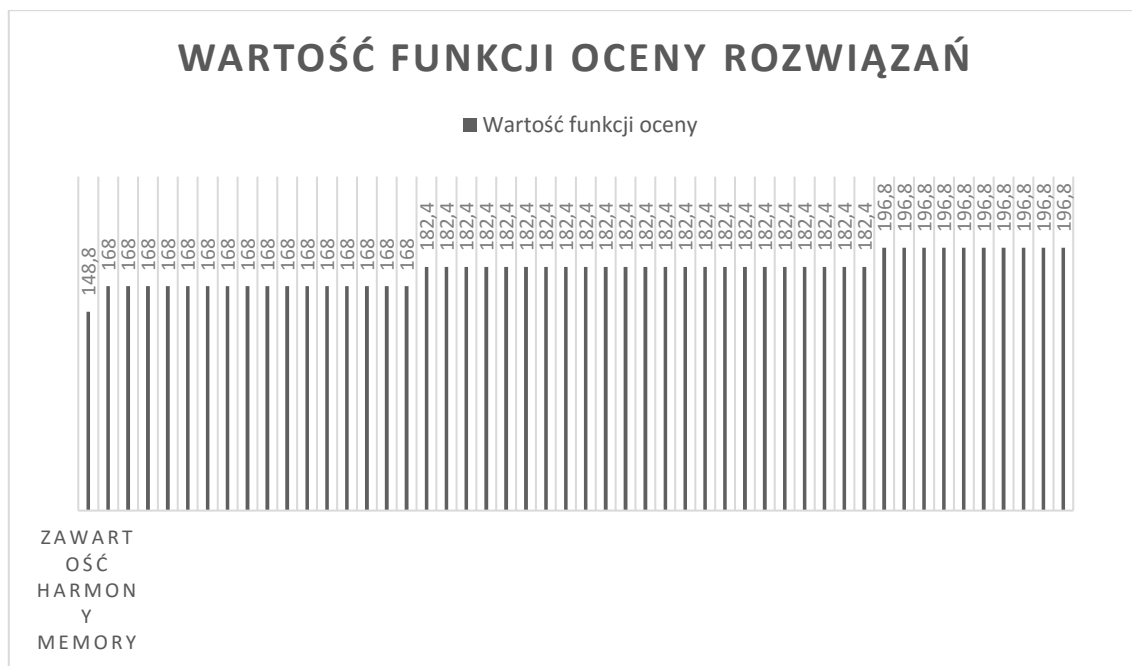
- Ocena wybranego rozwiązania:

	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
Harmoniczne współbrzmienie akompaniamentu z melodią główną	5	4	5	5	5
Rytm zmieniania się akordów	5	5	4	5	4
Podobieństwo akompaniamentu do melodii wejściowej	4	4	4	4	5
Różnorodność współbrzmień	5	4	5	5	4
<b>Ogólna ocena</b>	<b>5</b>	<b>4</b>	<b>4</b>	<b>5</b>	<b>5</b>

Tabela 5.5 Ocena testu wydajnościowego dla Hymnu Francji

## 6. Hymn Wielkiej Brytanii (główna melodia)

- Zróżnicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.11 Hymn Wielkiej Brytanii - zróżnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.12 Hymn Wielkiej Brytanii - progres najlepszego wyniku w iteracjach

- Ocena wybranego rozwiązania:

	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
--	---------	---------	---------	---------	---------

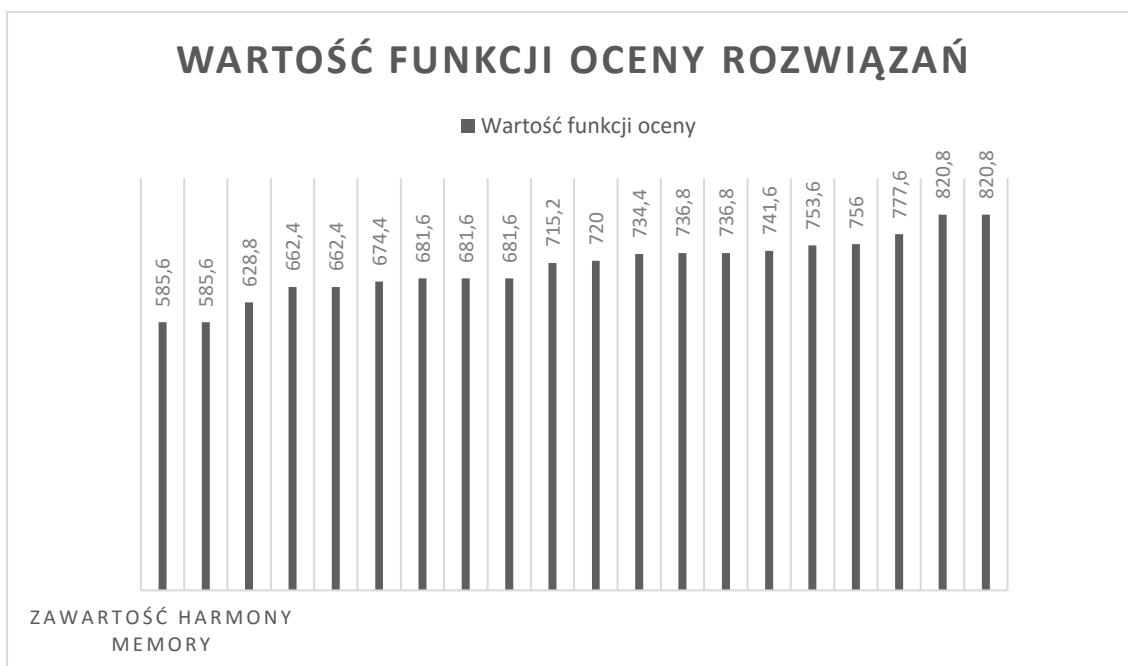
Harmoniczne współbrznienie akompaniamentu z melodią główną	3	3	2	3	3
Rytm zmieniania się akordów	5	4	4	4	4
Podobieństwo akompaniamentu do melodii wejściowej	3	3	2	3	4
Różnorodność współbrzmień	5	4	4	4	4
<b>Ogólna ocena</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>3</b>	<b>4</b>

Tabela 5.6 Ocena testu wydajnościowego dla Hymnu Wielkiej Brytanii

### 5.3.2.II wariant – testy jakości przy zachowaniu wydajności

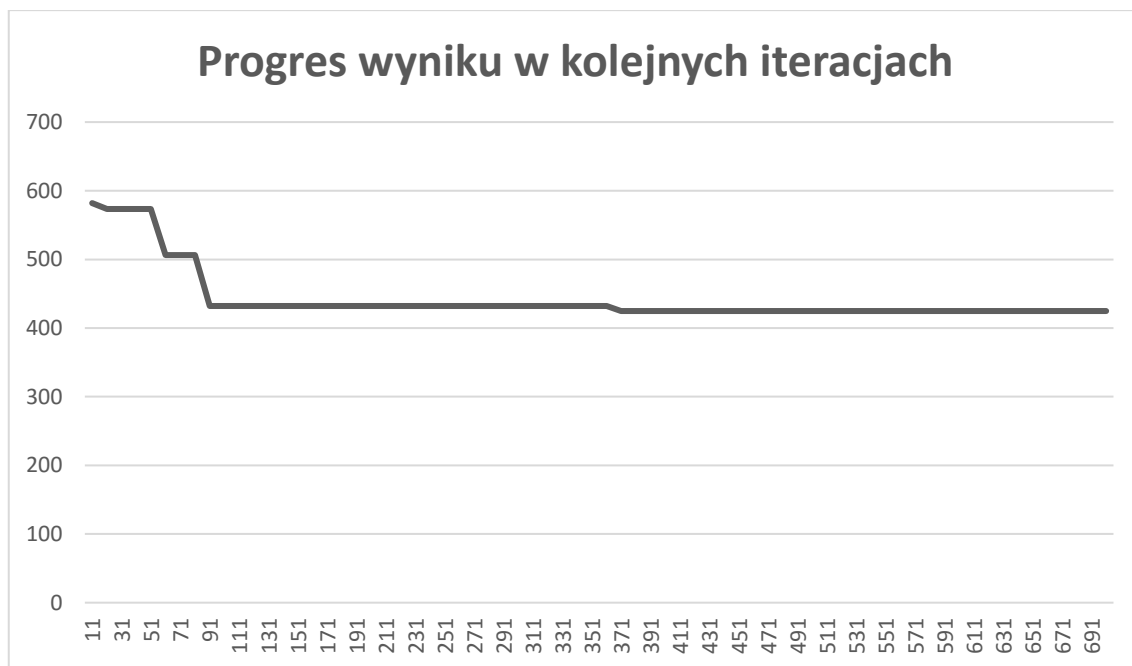
#### 1. Oberek powiślański (dwugłos: sopran i bas):

- Zróżnicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.13 Oberek Powiślański - zróżnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.14 Oberek powiślański - progres najlepszego wyniku w iteracjach

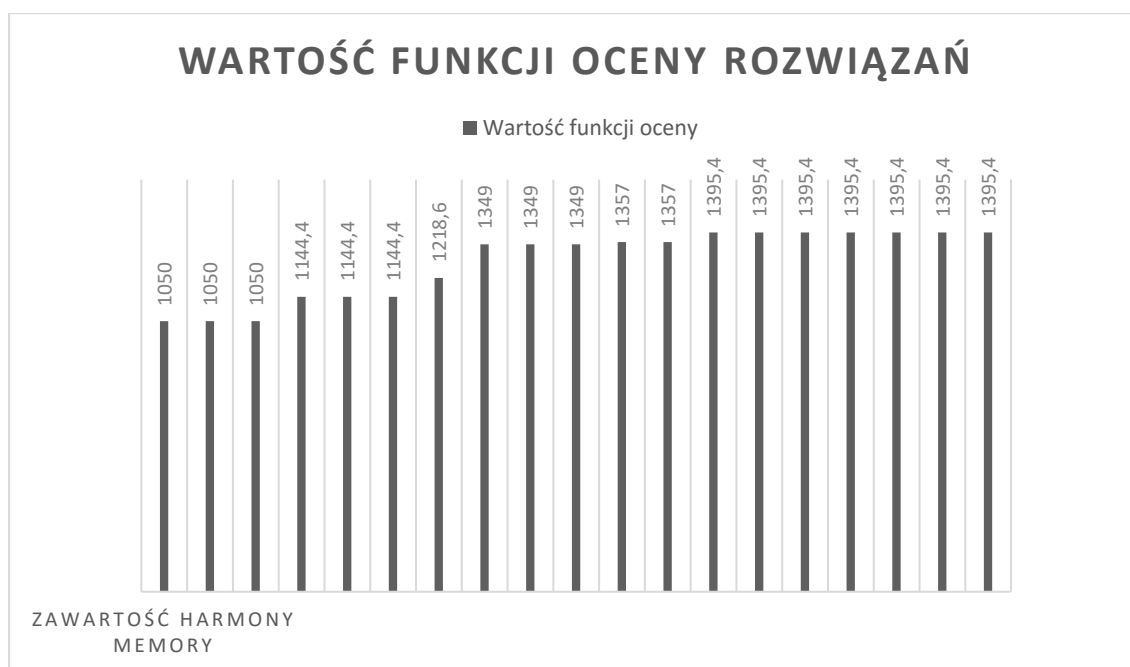
- Ocena wybranego rozwiązania:

	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
Harmoniczne współbrzmienie akompaniamentu z melodią główną	4	3	4	3	4
Rytm zmieniania się akordów	3	2	3	2	3
Podobieństwo akompaniamentu do melodii wejściowej	3	2	3	2	3
Różnorodność współbrzmień	3	2	3	2	3
<b>Ogólna ocena</b>	<b>3</b>	<b>2</b>	<b>3</b>	<b>2</b>	<b>3</b>

Tabela 5.7 Ocena testu wydajnościowego dla Oberka Powiślańskiego

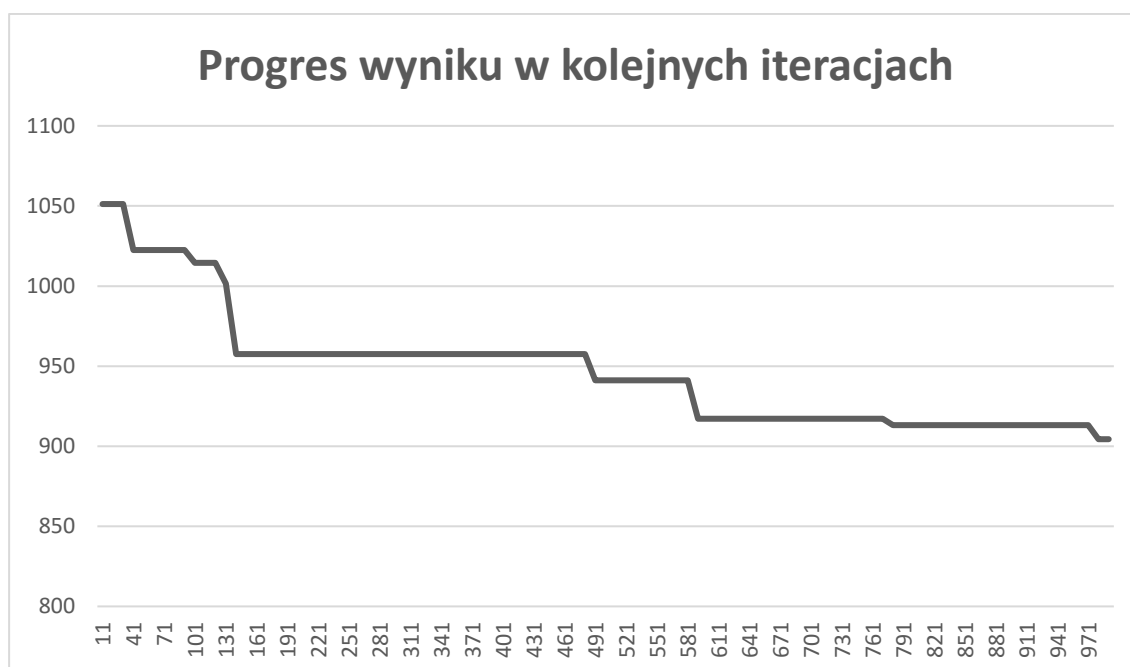
## 2. Hymn Włoch (melodie dwóch instrumentów):

- Zróznicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.15 Hymn Włoch - zróżnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.16 Hymn Włoch - progres najlepszego wyniku w iteracjach

- Ocena wybranego rozwiązania:

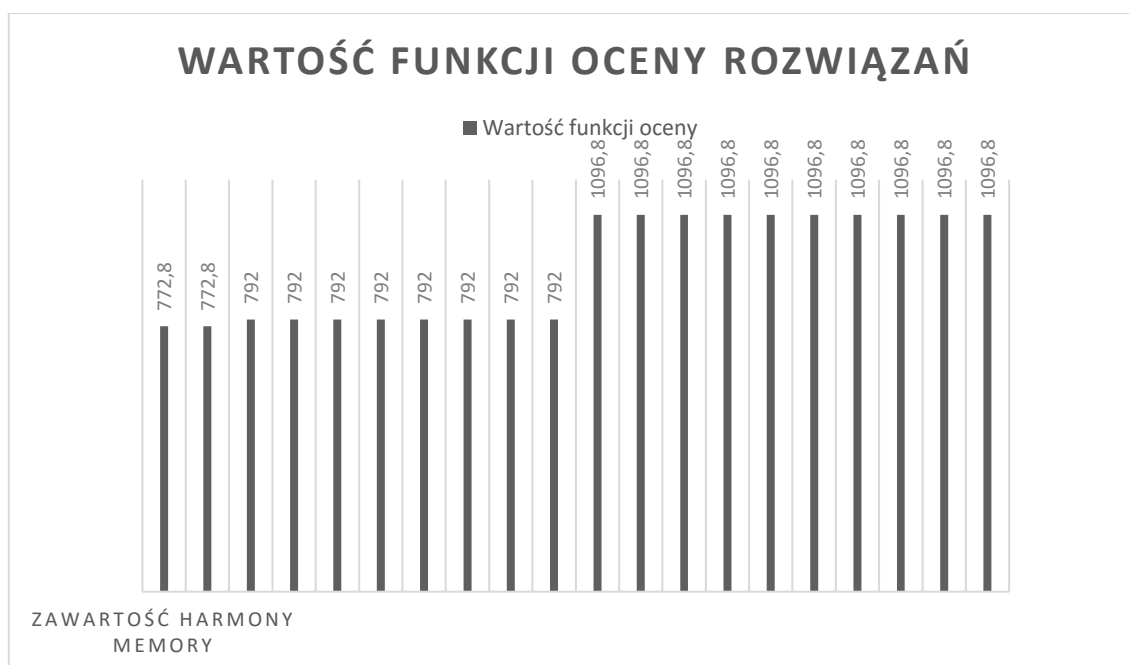
	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
--	---------	---------	---------	---------	---------

Harmoniczne współbrzmienie akompaniamentu z melodią główną	3	3	3	2	4
Rytm zmieniania się akordów	3	3	3	3	3
Podobieństwo akompaniamentu do melodii wejściowej	3	2	3	3	3
Różnorodność współbrzmień	4	3	3	2	3
<b>Ogólna ocena</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>2</b>	<b>3</b>

Tabela 5.8 Ocena testu wydajnościowego dla Hymnu Włoskiego

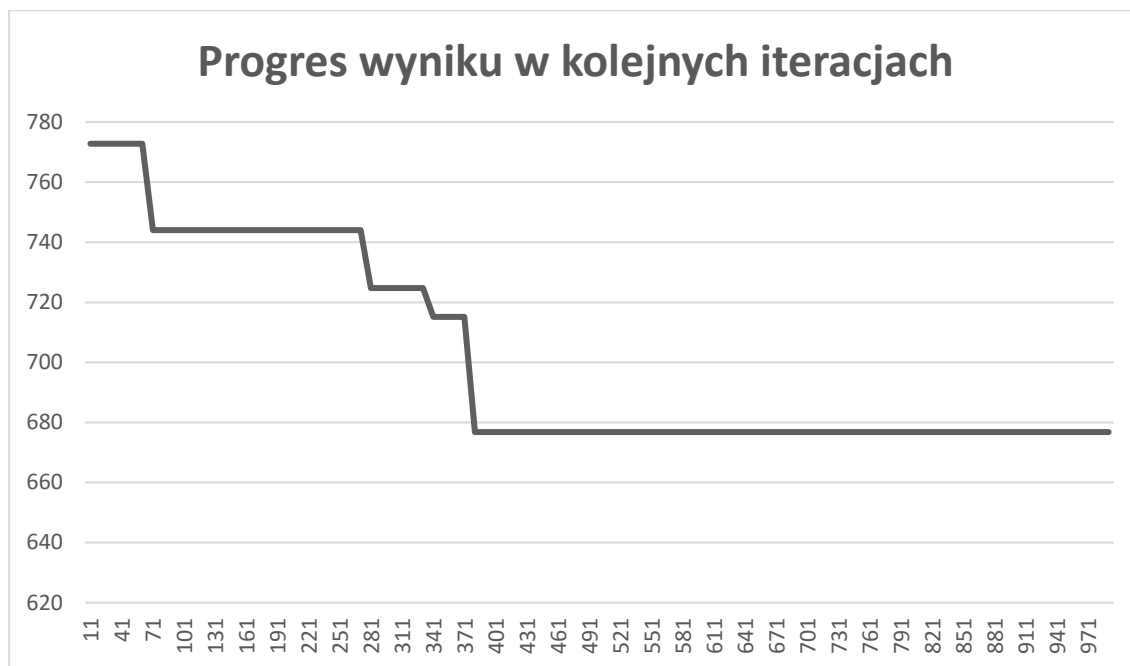
### 3. Hymn Stanów Zjednoczonych (melodie dwóch instrumentów):

- Zróźnicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.17 Hymn USA - zróźnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.18 Hymn USA - progres najlepszego wyniku w iteracjach

- Ocena wybranego rozwiązania:

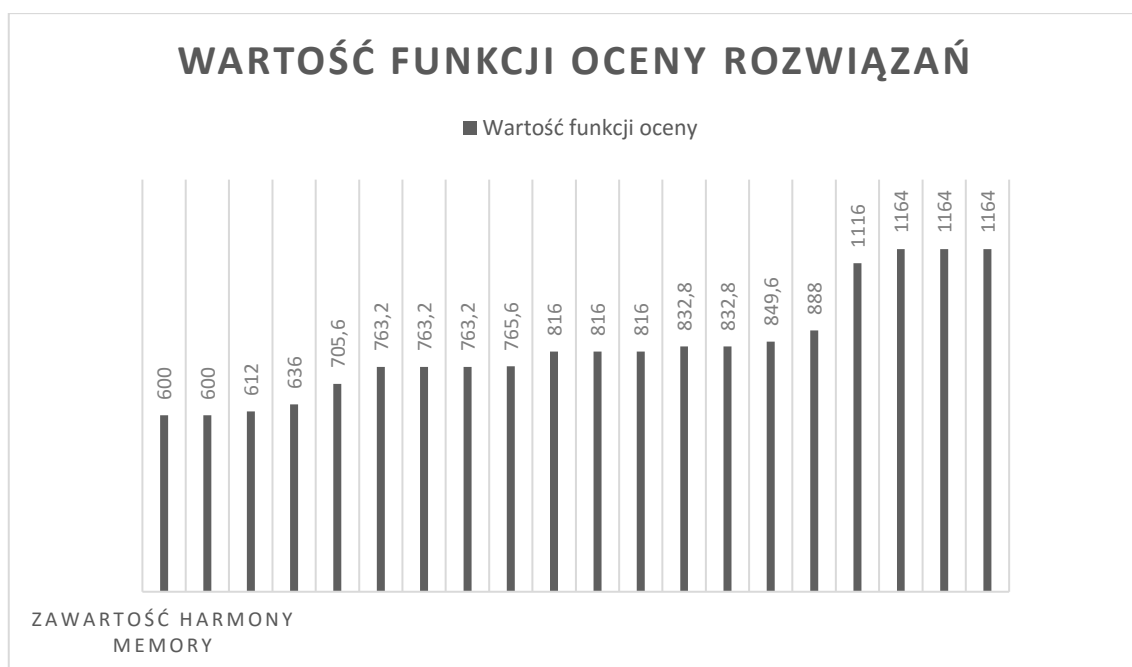
	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
Harmoniczne współbrzmienie akompaniamentu z melodią główną	4	3	3	4	4
Rytm zmieniania się akordów	3	3	2	3	3
Podobieństwo akompaniamentu do melodii wejściowej	4	3	3	4	3
Różnorodność współbrzmień	4	3	4	4	4
<b>Ogólna ocena</b>	<b>4</b>	<b>3</b>	<b>3</b>	<b>4</b>	<b>3</b>

Tabela 5.9 Ocena testu wydajnościowego dla Hymnu USA

#### 4. Główny motyw muzyczny z filmu „Piraci z Karaibów”

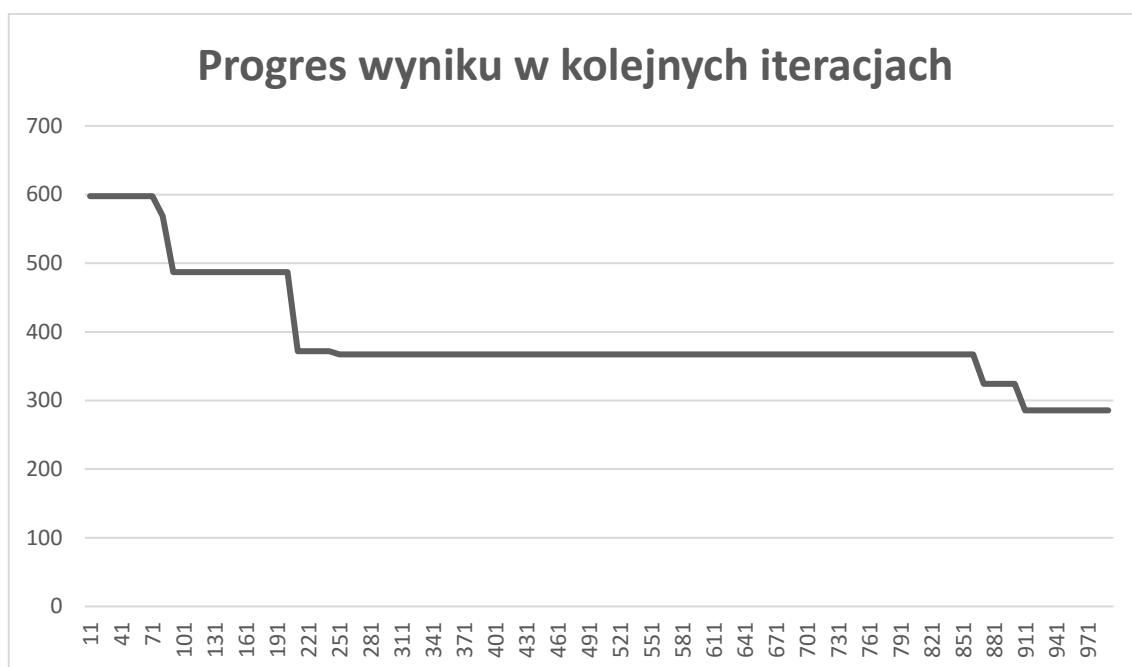
- Zróżnicowanie rozwiązań początkowych w pamięci harmonii:





Rysunek 5.19 Piraci z Karaibów - zróżnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.20 Piraci z Karaibów - progres najlepszego wyniku w iteracjach

- Ocena wybranego rozwiązania:

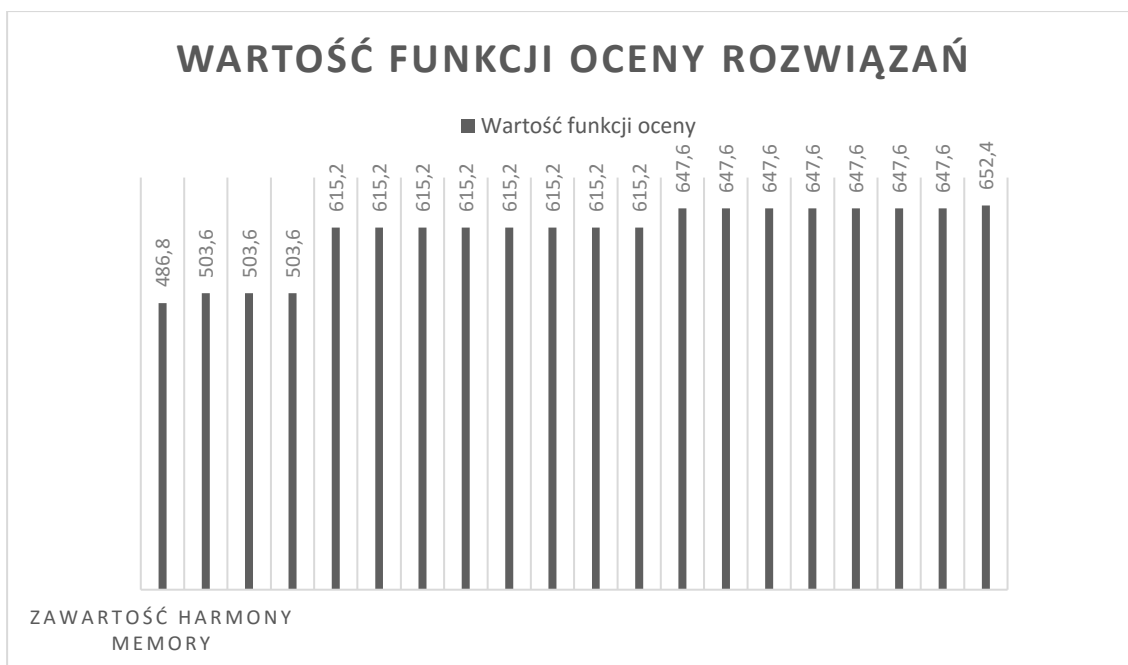
	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
--	---------	---------	---------	---------	---------

Harmoniczne współbrzmienie akompaniamentu z melodią główną	5	4	5	5	5
Rytm zmieniania się akordów	6	4	5	5	4
Podobieństwo akompaniamentu do melodii wejściowej	4	4	5	5	4
Różnorodność współbrzmień	5	4	4	6	5
<b>Ogólna ocena</b>	<b>5</b>	<b>4</b>	<b>5</b>	<b>5</b>	<b>4</b>

Tabela 5.10 Ocena testu wydajnościowego dla Piratów z Karaibów

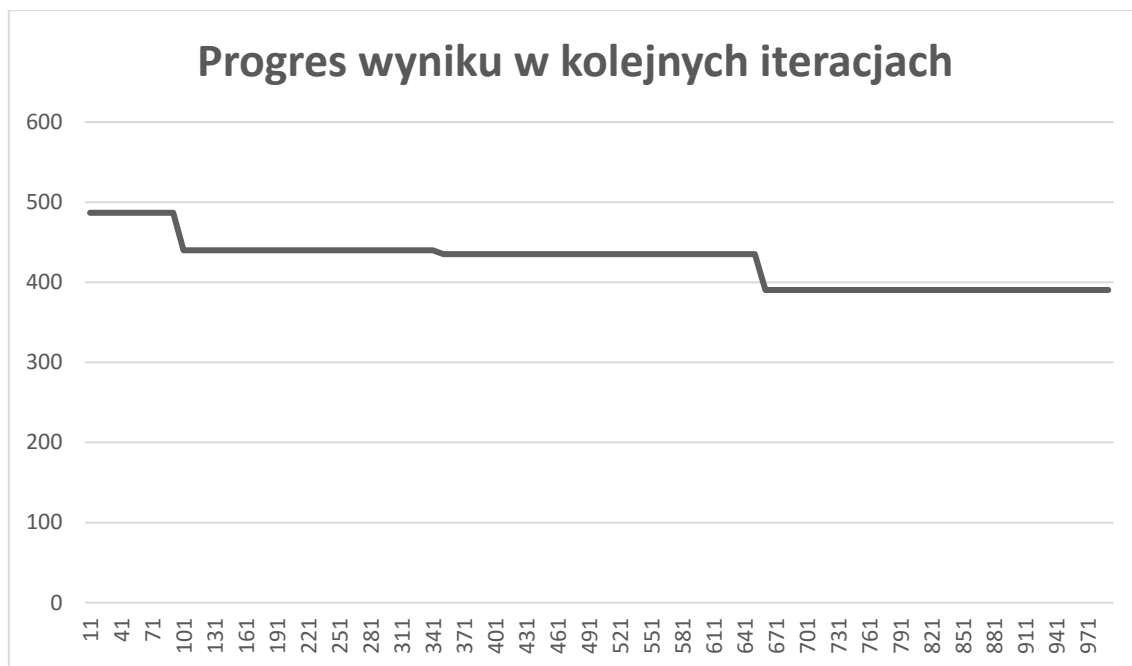
#### 5. Hymn Francji (główna melodia)

- Zróżnicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.21 Hymn Francji - zróżnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.22 Hymn Francji - progres najlepszego wyniku w iteracjach

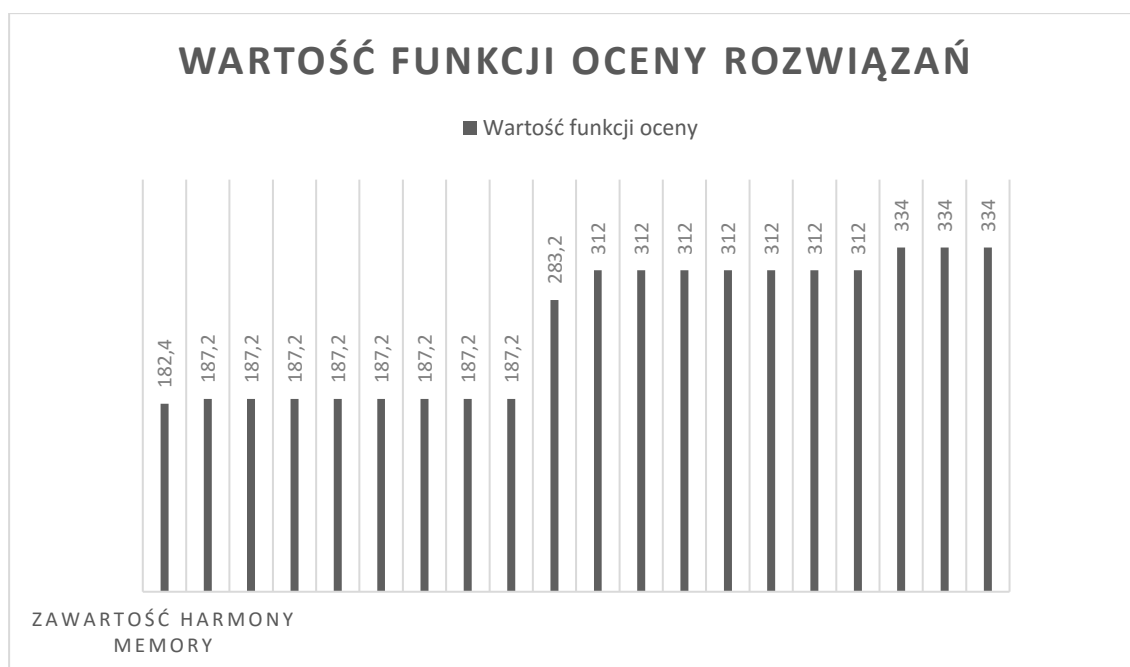
- Ocena wybranego rozwiązania:

	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
Harmoniczne współbrzmienie akompaniamentu z melodią główną	3	3	3	3	4
Rytm zmieniania się akordów	3	3	2	4	4
Podobieństwo akompaniamentu do melodii wejściowej	4	4	3	3	4
Różnorodność współbrzmień	3	3	3	4	3
<b>Ogólna ocena</b>	<b>3</b>	<b>3</b>	<b>2</b>	<b>3</b>	<b>4</b>

Tabela 5.11 Ocena testu wydajnościowego dla Hymnu Francji

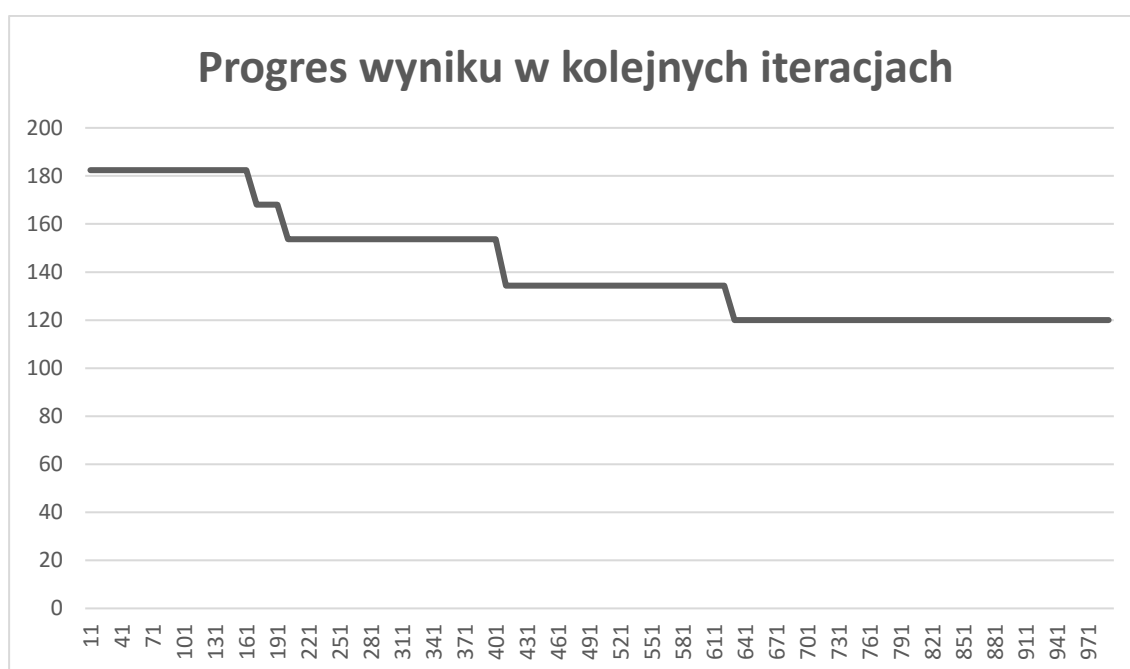
## 6. Hymn Wielkiej Brytanii (główna melodia)

- Zróżnicowanie rozwiązań początkowych w pamięci harmonii:



Rysunek 5.23 Hymn Wielkiej Brytanii - zróżnicowanie rozwiązań początkowych

- Progres wyniku najlepszego rozwiązania w kolejnych iteracjach:



Rysunek 5.24 Hymn Wielkiej Brytanii - progres najlepszego wyniku w iteracjach

- Ocena wybranego rozwiązania:

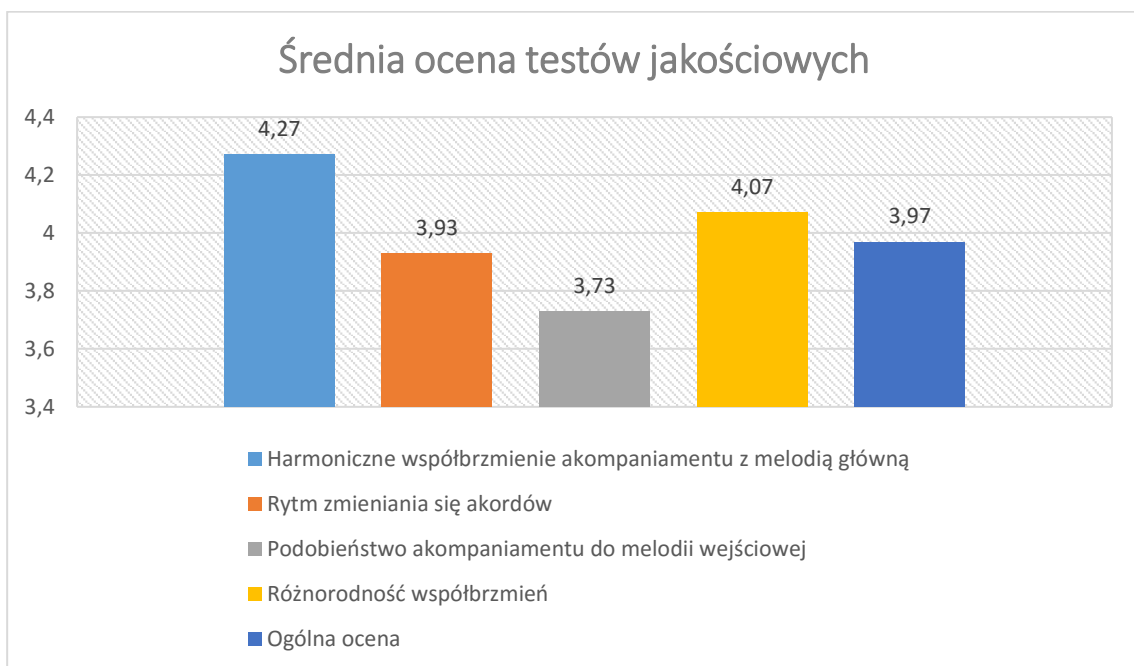
	Ocena 1	Ocena 2	Ocena 3	Ocena 4	Ocena 5
--	---------	---------	---------	---------	---------

Harmoniczne współbrzmienie akompaniamentu z melodią główną	3	2	2	3	3
Rytm zmieniania się akordów	2	3	3	2	2
Podobieństwo akompaniamentu do melodii wejściowej	4	2	2	3	4
Różnorodność współbrzmień	3	3	2	3	3
<b>Ogólna ocena</b>	<b>3</b>	<b>2</b>	<b>2</b>	<b>2</b>	<b>3</b>

Tabela 5.12 Ocena testu wydajnościowego dla Hymnu Wielkiej Brytanii

#### 5.4. Ocena jakości działania programu MusicAnalyzer

Z przeprowadzonych testów widać wyraźnie, iż ocena rozwiązań nie jest jednoznaczna, gdyż każdy z oceniających nadawał różne punkty w poszczególnych kategoriach. Jednakże po uśrednieniu wyników wszystkich jurorów i sprawdzanych plików wejściowych, można otrzymać statystykę, która umożliwia zbiorcze podsumowanie działania programu *MusicAnalyzer*.



Rysunek 5.25 Średnia ocena testów jakościowych MusicAnalyzer

Okazuje się, że najsilniejszą cechą powstającego akompaniamentu jest wrażenie jego zgodności harmoniczej z melodią wejściową, natomiast najslabszą ocena dużego stopnia identyczności obu melodii. Obie te cechy są ze sobą skorelowane, gdyż podobne miary określają obie te cechy w jednym przypadku pozytywnie a w innym negatywnie. Jest to spowodowane przede wszystkim faktem, iż program nie posiada szerszej bazy znanych melodii, przez co jest zmuszony opierać się jedynie na ścieżce wejściowej, przez co akompaniament jest do niej bardziej podobny. Pozytywnie zaś została oceniona różnorodność stosowanych współbrzmień,

natomiast nieco gorzej rytmika zmiana akordów. Ogólna ocena końcowa plasowała się w okolicy 4 w skali 1-6, co można uznać za wynik zadowalający. Widać zatem, iż stosowane metody oceny harmonii były trafne, natomiast brak rozbudowanej oceny rytmiczności układu w efekcie skutkowało często sztywnym, nieprzyjemnym rytmem zmian akordów. Najczęstszym warunkiem stopu okazał się w przypadku tego wariantu brak poprawy najlepszego wyniku przez określoną liczbę iteracji, choć w niektórych przypadkach pojawiał się on po bardzo licznych nieudanych próbach.

Drugi wariant testów opierał się na wygenerowaniu możliwie najlepszego rozwiązania przy mocno ograniczonych zasobach. Uśrednione wyniki ocen w poszczególnych kategoriach prezentują się następująco:



Rysunek 5.26 Średnia ocena testów wydajnościowych programu MusicAnalyzer

Zgodnie z przewidywaniami, rozwiązania z dużym ograniczeniem zasobów algorytmu zostały wyraźnie gorzej oceniane. Względem testów pełnej jakości, największy spadek w ocenie uzyskała harmonia brzmienia, różnorodność współbrzmień i ocena ogólna. Podobieństwo do melodii wejściowej oraz rytmiczność akordów nie zanotowały takich spadków, gdyż poprawa ich oceny w liczniejszych zbiorach i dłuższych iteracjach nie jest tak znaczna. W przypadku tak dobranych ograniczeń, najczęstszym warunkiem stopu algorytmu był limit czasu (3 sekundy), lecz dla krótszych kompozycji był to też limit 1000 iteracji. Oceny uzyskane przez program przy tej konfiguracji pozwalają przypuszczać, iż nie przynosiłby zadowalających rezultatów dla użytkownika przy tak surowych ograniczeniach. Pozytywnym aspektem pogorszenia się wyników względem wariantu I jest fakt, iż jest to dowód na to, że zastosowana implementacja algorytmu *Harmony Search* jest w stanie wyraźnie poprawić wygenerowane rozwiązania początkowe.

## 6. Podsumowanie

Automatyczne komponowanie muzyki jest problemem złożonym, trudnym w opracowaniu satysfakcjonującego rozwiązania i jego ocenienia, gdyż ostatecznie wyniki jego działania są poddawane ocenie estetycznej, nie do końca mierzalnej. Zastosowanie algorytmu *Harmony Search* sprostало pokładanym w nim oczekiwaniom, gdyż pozwalało znacznie polepszyć generowane rozwiązania początkowe w kolejnych iteracjach. Jednakże, należy sądzić, iż jego rezultaty nie spełniłyby wymagań stawianych prawdziwym kompozytorom i ich dziełom. Głównym ograniczeniem wynikającym z przyjętych założeń projektu był brak wcześniejszej bazy kompozycji. Algorytm musiał tworzyć dopasowanie do linii melodycznej korzystając jedynie z powszechnych zasad harmonii i rytmiki. Nie mniej jednak wyniki przeprowadzonych testów pozwalają sądzić, że program *MusicAnalyzer* mógłby wspomagać naukę kompozycji muzycznych amatorów poprzez znajdowanie harmonii układów muzycznych. Harmonia powstających kompozycji, budowana dzięki zasadom muzyki klasycznej, była wysoko oceniana przez osoby testujące, zatem można by wykorzystać ją w bardziej złożonym projekcie. Sam program jest bardzo uproszczonym wykorzystaniem algorytmu *HS* i nie daje możliwości utworzenia kompletnej kompozycji. Natomiast zawarty w nim model obiektów ze świata muzycznego oraz algorytm *Harmony Search* można wzbogacić o bazę znanych melodii prowadzących. Wówczas przy wykorzystaniu uczenia maszynowego można by tworzyć nie tylko podstawę harmoniczną lecz także własne melodie wiodące.

## 7. Wykaz literatury

W opracowaniu korzystano z następujących książek i artykułów:

1. A. Poszowski, *Harmonia systemu tonalnego dur-moll*, Wydaw. Akademii Muzycznej w Gdańsku, Gdańsk 2001
2. *Zanim zaczniesz grać na... : ABC muzyki*, Polskie Wydawnictwo Muzyczne, Kraków 2002
3. Andrzej Rakowski, *Percepcja wysokości dźwięku*, Państwowa Wyższa Szkoła Muzyczna, Warszawa 1978
4. Irena Poniatowska, *Dzieło muzyczne, teoria, historia, interpretacja*, PWM, Kraków 1984
5. Piotr Kałużny, *Skale muzyczne we współczesnej harmonii tonalnej*, Wydawnictwo Akademii Muzycznej, Poznań 1994
6. Jacek Targosz, *Podstawy harmonii funkcyjnej*, PWM, Kraków 2011
7. Witold Rudziński, *Nauka o rytmie muzycznym. C 1*, PWM, Kraków 1987
8. A. Długosz, *Kształcenie głosu. Śpiew solowy i chóralny. Jak uczyć się i uczyć innych*, nakład własny, Warszawa 1936
9. Z. W. Geem, J. H. Kim, *A New Heuristic Optimization Algorithm: Harmony Search*, 2001
10. M. G.H. Omran, , M. Mahdavi, *Global-best harmony search*, *Applied Mathematics and Computation* (2007)
11. K. S. Lee, Z. W. Geem, *A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice*, *Computational Methods Applied Mechanics Engineering* 194 (2005)
12. M. Mahdavi, M. Fesanghary, E. Damangir, *An improved harmony search algorithm for solving optimization problems*, *Applied Mathematics and Computation* 188 (2007)
13. D. Weyland, *A Rigorous Analysis of the Harmony Search Algorithm: How the Research Community can be Misled by a "Novel" Methodology*, *International Journal of Applied Metaheuristic Computing*, 1 (2010)
14. W. Rudziński, *Muzyka dla wszystkich : z licznymi przykładami, rysunkami i tablicami*, PWM, Kraków 1948
15. Franciszek Wesolowski, *Zasady muzyki : podręcznik dla średnich szkół muzycznych*, PWM, Kraków 1980
16. K. Sikorski, *Harmonia cz. 2*, Polskie Wydawnictwo Muzyczne, Kraków 1948
17. Michał Zieliński, *Technika orkiestrowa w utworach Karola Szymanowskiego*, Wydawnictwo uczelniane AM., Bydgoszcz 1993
18. Elżbieta Sławińska, *Przetwarzanie dźwięku przez ucho*, Państwowa Wyższa Szkoła Muzyczna, Warszawa 1968
19. Barbara Smoleńska-Zielińska, *Przeżycie estetyczne muzyki*, WSiP, Warszawa 1991

Podczas tworzenia pracy korzystano także z następujących artykułów i źródeł internetowych:



20. <https://pl.wikipedia.org/wiki/MIDI>, (dostęp 14.09.2016)
21. G. Johnson, *Undiscovered Bach? No, a Computer Wrote It*, 11.11.1997, <http://www.nytimes.com/1997/11/11/science/undiscovered-bach-no-a-computer-wrote-it.html?pagewanted=all>, (dostęp 20.08.2016)
22. <http://artsites.ucsc.edu/faculty/cope/Emily-howell.htm>, (dostęp 20.08.2016)
23. Jacqui Cheng, *Virtual composer makes beautiful music - and stirs controversy*, 30.09.2009, <http://arstechnica.com/science/2009/09/virtual-composer-makes-beautiful-musicand-stirs-controversy/>, (dostęp 20.08.2016)
24. <https://www.google.com/patents/US7696426>, (dostęp 22.08.2016)
25. D. L. Brown, *Expressing narrative function in adaptive, computer-composed music*, 06.2012, [http://www.danielbrownmusic.com/uploads/1/3/2/3/13234393/final\\_dissertation\\_final\\_edit.pdf](http://www.danielbrownmusic.com/uploads/1/3/2/3/13234393/final_dissertation_final_edit.pdf), (dostęp 22.08.2016)
26. <http://www.athtek.com/digiband/music-composition-software.html>, (dostęp 1.09.2016)
27. J. P. Bello, MIDI Code, [https://www.nyu.edu/classes/bello/FMT\\_files/9\\_MIDI\\_code.pdf](https://www.nyu.edu/classes/bello/FMT_files/9_MIDI_code.pdf) (dostęp 14.09.2016)
28. E. Szlachcic, *Teoria i metody optymalizacji*, [http://staff.iiar.pwr.wroc.pl/ewa.szlachcic/materialy%20dydaktyczne/air\\_studia\\_2\\_stopnia/12\\_w\\_timo\\_optymalizacja\\_globalna.pdf](http://staff.iiar.pwr.wroc.pl/ewa.szlachcic/materialy%20dydaktyczne/air_studia_2_stopnia/12_w_timo_optymalizacja_globalna.pdf), (dostęp 20.09.2016)
29. <http://www.k0pper.republika.pl/geny.htm>, (dostęp 20.09.2016)
30. <http://www.codeproject.com/Articles/339416/Printing-large-WPF-UserControls>, (dostęp 08.09.2016)
31. <https://www.midi.org/specifications/item/gm-level-1-sound-set>, (dostęp 14.09.2016)
32. L. Sanford, *C# MIDI Toolkit*, 18.04.2007, <http://www.codeproject.com/Articles/6228/C-MIDI-Toolkit>, dostęp 14.09.2016
33. Jacek Salamon, *PSAM Control Library*, 24.06.2010, <http://www.codeproject.com/Articles/87329/PSAM-Control-Library>, dostęp 22.03.2016
34. Jacek Salamon, *PSAM WPF Control Library*, 24.06.2010, <http://www.codeproject.com/Articles/89582/PSAM-WPF-Control-Library>, dostęp 22.03.2016

## 8. Wykaz rysunków

Rysunek 3.1 Dźwięki gamy C-dur, źródło <a href="http://3.bp.blogspot.com/-ZbOavfTrYJg/T1QALqWFwdI/AAAAAAAAADg/eC5xMCH9c6g/s1600/gama-c-dur-odleglosci.png">http://3.bp.blogspot.com/-ZbOavfTrYJg/T1QALqWFwdI/AAAAAAAAADg/eC5xMCH9c6g/s1600/gama-c-dur-odleglosci.png</a> .....	15
Rysunek 3.2 Układ klawiszy fortepianu i odpowiadających im dźwięków na pięcioliniach, źródło: <a href="https://tucsonsongstress.files.wordpress.com/2012/11/staffnkeys.gif">https://tucsonsongstress.files.wordpress.com/2012/11/staffnkeys.gif</a> .....	16
Rysunek 3.3 Klucz wiolinowy, źródło <a href="http://www.music-paper.com/ImagesClefs/treble-clef.jpg">http://www.music-paper.com/ImagesClefs/treble-clef.jpg</a>	16
Rysunek 3.4 Klucz basowy, źródło: <a href="http://www.theyrenotourgoats.com/wp-content/uploads/2014/10/Bass_Clef.jpg">http://www.theyrenotourgoats.com/wp-content/uploads/2014/10/Bass_Clef.jpg</a> .....	16
Rysunek 3.5 Skala molowa eolska w gamie na przykładzie gamy a – moll, źródło: <a href="http://3.bp.blogspot.com/-c1_q-799WgM/T1QOY7uNJGI/AAAAAAAAACjI/hpUDBezDOs4/s400/gama-a-moll-odleglosci.png">http://3.bp.blogspot.com/-c1_q-799WgM/T1QOY7uNJGI/AAAAAAAAACjI/hpUDBezDOs4/s400/gama-a-moll-odleglosci.png</a> ....	18
Rysunek 3.6 Skala harmoniczna na przykładzie gamy a - moll, źródło: <a href="https://upload.wikimedia.org/wikipedia/commons/thumb/4/4b/Moll_harm.jpg/600px-Moll_harm.jpg">https://upload.wikimedia.org/wikipedia/commons/thumb/4/4b/Moll_harm.jpg/600px-Moll_harm.jpg</a> .....	18
Rysunek 3.7 Dźwięki należące do gamy C - dur zapisane w kluczu wiolinowym .....	19
Rysunek 3.8 Akord C-dur w przewrotach, źródło: <a href="http://www.cyja.webd.pl/akordeon/images/stories/15-01%20przewroty%20c.jpg">http://www.cyja.webd.pl/akordeon/images/stories/15-01%20przewroty%20c.jpg</a> .....	20
Rysunek 3.9 Akordy triady harmonicznego w gamie C - dur .....	21
Rysunek 3.10 Znaki przy kluczu wiolinowym, źródło: <a href="https://upload.wikimedia.org/wikipedia/commons/thumb/8/84/A-major_f-sharp-minor.svg/150px-A-major_f-sharp-minor.svg.png">https://upload.wikimedia.org/wikipedia/commons/thumb/8/84/A-major_f-sharp-minor.svg/150px-A-major_f-sharp-minor.svg.png</a> .....	21
Rysunek 3.11 Przykłady metrum i ich wyjaśnienie, źródło: <a href="http://e-perkusja.pl/plik/nauka_gry_na_perkusji/Artykuly/Czytanie_z_nut/04_Metrum_gorna_cyfra.jpg">http://e-perkusja.pl/plik/nauka_gry_na_perkusji/Artykuly/Czytanie_z_nut/04_Metrum_gorna_cyfra.jpg</a>	22
Rysunek 3.12 Schmeta działania Emily Howell, źródło <a href="https://www.google.com/patents/US7696426">https://www.google.com/patents/US7696426</a> .....	24
Rysunek 3.13 Schemat działania programu Mezzo, źródło <a href="http://www.danielbrownmusic.com/uploads/1/3/2/3/13234393/final_dissertation_final_edit.pdf">http://www.danielbrownmusic.com/uploads/1/3/2/3/13234393/final_dissertation_final_edit.pdf</a>	26
Rysunek 3.14 Interfejs programu DigiBand, źródło <a href="http://www.athtek.com/image/digiband/composition.jpg">http://www.athtek.com/image/digiband/composition.jpg</a> , strona odwiedzona 1.09.2016 .....	27
Rysunek 4.1 Widok początkowy okna StartWindow w programie MusicAnalyzer .....	28
Rysunek 4.2 Widok drugiej zakładki w oknie StartWindow w programie MusicAnalyzer .....	29
Rysunek 4.3 Widok trzeciej zakładki w oknie StartWindow w programie MusicAnalyzer .....	29
Rysunek 4.4 Widok głównego okna PlayerWindow w programie MusicAnalyzer .....	30
Rysunek 4.5 Widok opcji Menu w oknie PlayerWindow w programie MusicAnalyzer .....	32
Rysunek 4.6 Komunikat o błędzie pliku wejściowego w programie MusicAnalyzer .....	32
Rysunek 4.7 Komunikat o braku plików konfiguracyjnych w MusicAnalyzer .....	33
Rysunek 4.8 Fragment zapisu nutowego pieśni "Kozak" S. Moniuszki z akompaniamentem pianina (c-moll). .....	36

Rysunek 4.9 Tonacje i znaki przy kluczu wiolinowym, źródło <a href="http://blog.gitarius.pl/kurs_gitarowy/chro4.png">http://blog.gitarius.pl/kurs_gitarowy/chro4.png</a> .....	37
Rysunek 4.10 Podział wartości rytmicznych, źródło <a href="http://whittier.mpls.k12.mn.us/uploads/rhythm_tree_1.jpg">http://whittier.mpls.k12.mn.us/uploads/rhythm_tree_1.jpg</a> .....	38
Rysunek 4.11 Fragment partii tenora z chóralnej aranżacji utworu ludowego "Oberek powiślański" .....	40
Rysunek 4.12 Trójdźwięk zapisany na pięciolinii, źródło <a href="http://www.codeproject.com/KB/miscctrl/psamcontrollibrary/examplechords.png">http://www.codeproject.com/KB/miscctrl/psamcontrollibrary/examplechords.png</a> , strona odwiedzona 1.09.2016 .....	41
Rysunek 4.13 Fragment zapisu nutowego z nutami o różnym kierunku .....	41
Rysunek 4.14 Przykładowy zapis nutowy dwóch głosów w programie MusicAnalyzer .....	42
Rysunek 4.15 Struktura pamięci rozwiązań (Harmony memory), źródło: <a href="http://s3.amazonaws.com/academia.edu.documents/2309202/j_2001_simulation.pdf">http://s3.amazonaws.com/academia.edu.documents/2309202/j_2001_simulation.pdf</a> .....	43
Rysunek 4.16 Jednoczesne trwanie dźwięków, źródło: <a href="http://www.musanim.com/fugueplaying/fugueplaying.gif">http://www.musanim.com/fugueplaying/fugueplaying.gif</a> .....	45
Rysunek 4.17 Fragment III Koncertu Fortepianowego S. Rachmaninowa, źródło: <a href="https://upload.wikimedia.org/wikipedia/commons/thumb/d/db/Piano_Concerto_3_Cadenza.png/350px-Piano_Concerto_3_Cadenza.png">https://upload.wikimedia.org/wikipedia/commons/thumb/d/db/Piano_Concerto_3_Cadenza.png/350px-Piano_Concerto_3_Cadenza.png</a> .....	56
Rysunek 4.18 Przykładowy wydruk zapisu nutowego akompaniamentu.....	57
Rysunek 5.1 Oberek Powiślański - zróżnicowanie rozwiązań początkowych .....	63
Rysunek 5.2 Oberek powiślański - progres najlepszego wyniku w iteracjach .....	64
Rysunek 5.3 Hymn Włoch - zróżnicowanie rozwiązań początkowych.....	65
Rysunek 5.4 Hymn Włoch - progres najlepszego wyniku w iteracjach.....	65
Rysunek 5.5 Hymn USA - zróżnicowanie rozwiązań początkowych .....	66
Rysunek 5.6 Hymn USA - progres najlepszego wyniku w iteracjach .....	67
Rysunek 5.7 Piraci z Karaibów - zróżnicowanie rozwiązań początkowych .....	68
Rysunek 5.8 Piraci z Karaibów - progres najlepszego wyniku w iteracjach .....	68
Rysunek 5.9 Hymn Francji - zróżnicowanie rozwiązań początkowych .....	69
Rysunek 5.10 Hymn Francji - progres najlepszego wyniku w iteracjach .....	70
Rysunek 5.11 Hymn Wielkiej Brytanii - zróżnicowanie rozwiązań początkowych .....	71
Rysunek 5.12 Hymn Wielkiej Brytanii - progres najlepszego wyniku w iteracjach .....	71
Rysunek 5.13 Oberek Powiślański - zróżnicowanie rozwiązań początkowych .....	72
Rysunek 5.14 Oberek powiślański - progres najlepszego wyniku w iteracjach .....	73
Rysunek 5.15 Hymn Włoch - zróżnicowanie rozwiązań początkowych.....	74
Rysunek 5.16 Hymn Włoch - progres najlepszego wyniku w iteracjach.....	74
Rysunek 5.17 Hymn USA - zróżnicowanie rozwiązań początkowych .....	75
Rysunek 5.18 Hymn USA - progres najlepszego wyniku w iteracjach .....	76
Rysunek 5.19 Piraci z Karaibów - zróżnicowanie rozwiązań początkowych .....	77
Rysunek 5.20 Piraci z Karaibów - progres najlepszego wyniku w iteracjach .....	77
Rysunek 5.21 Hymn Francji - zróżnicowanie rozwiązań początkowych.....	78

Rysunek 5.22 Hymn Francji - progres najlepszego wyniku w iteracjach .....	79
Rysunek 5.23 Hymn Wielkiej Brytanii - zróżnicowanie rozwiązań początkowych .....	80
Rysunek 5.24 Hymn Wielkiej Brytanii - progres najlepszego wyniku w iteracjach.....	80
Rysunek 5.25 Średnia ocena testów jakościowych MusicAnalyzer .....	81
Rysunek 5.26 Średnia ocena testów wydajnościowych programu MusicAnalyzer .....	82

## 9. Wykaz tabel

Tabela 4.1 Wartości rytmiczne przedłużane, źródło <a href="https://musictheorysite.files.wordpress.com/2015/06/dotted-value.png">https://musictheorysite.files.wordpress.com/2015/06/dotted-value.png</a> .....	39
Tabela 4.2 Rozkład akcentów w takcie w zależności od metrum utworu .....	48
Tabela 4.3 Akordy zastępcze podstawowych funkcji harmoniczych .....	50
Tabela 4.4 Możliwe akordy septymowe .....	50
Tabela 4.5 Przykładowe zróżnicowanie rozwiązań początkowych .....	51
Tabela 4.6 Zestaw akordów alternatywnych dla procesu improwizacji .....	52
Tabela 4.7 Progres wyniku najlepszego rozwiązania w iteracjach algorytmu .....	53
Tabela 5.1 Ocena testu wydajnościowego dla Oberka Powiślańskiego .....	64
Tabela 5.2 Ocena testu wydajnościowego dla Hymnu Włoskiego .....	66
Tabela 5.3 Ocena testu wydajnościowego dla Hymnu USA .....	67
Tabela 5.4 Ocena testu wydajnościowego dla Piratów z Karaibów .....	69
Tabela 5.5 Ocena testu wydajnościowego dla Hymnu Francji .....	70
Tabela 5.6 Ocena testu wydajnościowego dla Hymnu Wielkiej Brytanii .....	72
Tabela 5.7 Ocena testu wydajnościowego dla Oberka Powiślańskiego .....	73
Tabela 5.8 Ocena testu wydajnościowego dla Hymnu Włoskiego .....	75
Tabela 5.9 Ocena testu wydajnościowego dla Hymnu USA .....	76
Tabela 5.10 Ocena testu wydajnościowego dla Piratów z Karaibów .....	78
Tabela 5.11 Ocena testu wydajnościowego dla Hymnu Francji .....	79
Tabela 5.12 Ocena testu wydajnościowego dla Hymnu Wielkiej Brytanii .....	81