3C7-2

木構造型遺伝子を用いたクラシック音楽作曲支援システム:CACIE

CACIE: Classical Piece Composition Aid System by means of Tree Gene Reperesentation

安藤 大地[*1] 伊庭 斉志[*1]
Daichi Ando Hitoshi Iba

[*1]東京大学大学院新領域創成科学研究科
Graduate School of Frontier Sciences, The University of Tokyo

Research on the application of Interactive Evolutionary Computation (IEC) to the field of musical composition has been improved in recent years, marking an interesting parallel to the current trend of applying human characters or sensitivities to computers systems. However, past techniques developed for the IEC-based composition have not necessarily proven very effective for the sake of professional use. This is due to the large deference between data representation used by IEC and authorized classic music composition. To solve these difficulties, we purpose a new IEC approach to music composition based on the classical music theory. In this paper, we describe an established system according to the above idea, and how successfully we can compose a piece using the system.

1. はじめに

計算機の発展とともに用いられるようになってきた確率論的な作曲手法の利点は，作曲者自身も予想ができない結果を得ることにある．しかしながら従来の確率論的な作曲手法では，作曲者が計算機の出力を評価選択し，得られた結果をさらに修正する必要があった．

そこで Dawkins によって提案された対話型進化論的計算 [Dawkins 86] を，作曲支援に用いるシステムが提案された．対話型進化論的計算 (Interactive Evolutionary Computation, IEC) では，初期集団はユーザが定義した範囲内でランダムに生成されるが，ユーザとシステムの対話を通して徐々にユーザが求めるものへ収束していき，ユーザは出力結果として完全に満足できるものを得られる．また対話型進化論的計算の交配や突然変異などの遺伝子操作などは基本的にランダムに行われるため，ユーザが予想できなかったより良い結果を得る余地も多分に残されている．

進化論定計算の探索効率は，どのように問題を遺伝子として表現するか，どのような遺伝子操作を用いるかに大きく影響を受ける．つまり進化論的計算を作曲に応用する時は，作曲プロセスや楽曲をどのような問題としてとらえ，進化論的計算のメソッドにどのようにマッピングするかが重要になる．また，対話型であるため，ユーザインタフェースも効率に大きな影響を与える．

過去にも様々な音楽の遺伝子表現やユーザインタフェイスが試みられてきた．進化論的計算，特に遺伝的アルゴリズム (Genetic Algorithm, GA) と遺伝的プログラミング (Genetic Programming, GP) を用いた研究は [Burton 99] にまとめられている．

楽譜表現や音響を数式によって記述する方法 [Laine 94, Putnam 94]，GP のツリー構造によって音符やコードの接続の関係性を表現する [Johanson 98, Tokui 00] などが既存の研究とある．特に後者のツリー構造によって楽譜表現を記述する方法は，古典的な音楽学の分析作業でも用いられる．そのため理解が容易であり，手動による遺伝子の修正などの IEC の諸テクニックが用いやすいという利点がある．さらに，ツリー構造による楽譜表現をさらに発展させた，クラシック音楽に見られる典型的な繰り返し構造を表現する方法も提案されている．[Dahlstedt 04]

しかしながら，これらのツリー構造による遺伝子表現や遺伝子操作では，巨大な曲を生成しようとした時にツリー構造が非常に複雑になる．そのためユーザが提示された染色体を見た時に，生成される楽曲の構造の直感的な把握が難しくなる．また小さい集団サイズで複雑な染色体を扱おうとすると終息の効率が悪化する．これらの理由により，既存のシステムは，フレーズかメロディのような比較的短い音列の生成への適用が中心で，ユーザの負担を考えると，長くても小品程度の長さの曲しか合成できなかった．

このような過去の研究の成果と問題を踏まえ，筆者らは新たなシステム CACIE(Computer Aided Composition using Interactive Evolution) を構築した．CACIE は無町の現代音楽作品の作曲を支援する対話型 GP システムである．

2. 遺伝子表現

2.1 木構造による音楽構造の表現

CACIE では楽譜表現の遺伝子表現としてツリー構造を採用した．クラシック音楽の学習における楽曲の分析において，分析結果をツリー構造で表すことは多い．そのため，ツリー構造による音楽構造の表現は音楽家にとって理解しやすい表現手法であると考えられる．

ツリー構造による音楽表現では，非終端ノードの工夫により，様々な音楽的構造を簡易に表現することが可能である．その他にも非終端ノードを適当に用いることで，同一形態の表現で小規模なフレーズから大きな楽曲構造まで表現することができる．

図 1 にスコアとツリー構造の相互変換を示す．ここで非終端ノードの S は Sequence の略であり二つの引数を順番に演奏する関数である．同様に U は Unite の略であり二つの引数を同時に演奏する関数である．終端ノードとしては，一つ又は複数の音符からなるリストを取る．一つの音符には Note Number, Amplitude, Duration, OnsetTime の各要素が含まれている．また Amplitude が 0 の時は，その音符は休符として扱われる．また，Sequence と Unite 以外にも様々な関数が提供される．用意されている全ての関数は典型的なクラシック音楽で使用されている音楽構造を作曲家が理解しやすい形で実装したものである．関数として実装する音楽構造の選択にあたっては，二つ

連絡先: 安藤大地，東京大学，柏市柏の葉 5-1-5，04-7136-3873，dando@iba.k.u-tokyo.ac.jp

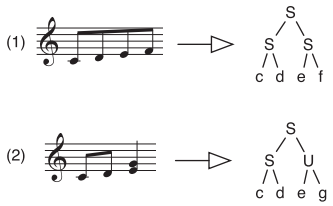

図 1: Tree representation of musical phrase

表 1: List of part of functions

S	Connect two arguments continuously (S a b) = (a b)
U	Connect two arguments simultaneously (U a b) = (ab)
SR	Make a repetition of two arguments (SR 5 a b) = (a b a b a)
D	Apply rhythm pattern of 2nd argument to 1st argument (D a(60,100,10) b(62,120,20)) = a(60,100,20)
P	Apply pittch pattern of 2nd argument to 1st argument (P a(60,100,10) b(62,120,20)) = a'(62,100,10)
A	Apply articulation pattern of 2nd argument to 1st argument (A a(60,100,10) b(62,120,20) = a'(60,120,10)
RV	Reverse ordering (RV (a b)) = (b a)
IV	Pitch Inversion (IV a((60,100,10)(62,120,20)) = a'((62,100,10)(60,120,20))
TP	Transpose (TP+5 a((60,100,10)(62,120,20)) = a'((65,100,10)(67,120,20))
MS	Return a sequence as follows: (MS (a b c) (d e)) = (a d b c e)
MU	Return a sequence as follows: (MU (a b) (c d)) = ((U a c) (U b d))
CAR	Return a sequence as follows: (CAR50\% (a b c d)) = (a b)
CDR	Return a sequence as follows: (CDR50\% (a b c d)) = (c d)
ACML	Return an accumulated sequence (ACML (a b c)) = (a a b a b c)
FILP	Return a sequence contains repetition with pitch transposing as follows: (FILP+2 (60 63) 65) = (60 63 62 65 64 67 65)

の終端ノードをどのように繋げるかという関係性を重視した．表 1 に実装された関数の一部を提示する．

これらの関数は再利用可能なライブラリとしても提供した．したがって，ユーザは，システムで提供された関数を組合せて新しい関数をプログラミングし新たにシステムに取り込むことが可能である．

2.2 再帰終端ノード

筆者らは，通常の終端，非終端ノードの他に，RecursiveTerminal Node という特殊な終端ノードを用意した．これにより染色体の肥大化を防ぎながら，効率的にクラシック音楽的な繰り返し構造を実現することができる．Recusive Terminal Node は自らの直上の非終端ノードを参照し，その部分木で自分を置き換える．同時にコピーされた部分木内の Recursive Terminal Node は，自身が保持する通常の終端ノードに置き換えられる．図 2，3 に展開例を示す．

2.3 遺伝子操作

交配は通常の部分木の交換と線形 GP の一点交差を採用した．また突然変異として，通常の個体内での部分木の交換の他に Increase と Decrease という特殊な操作を採用した．

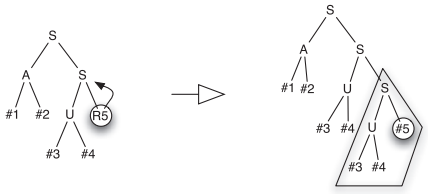

図 2: Developmental process of a Recursive Terminal Node

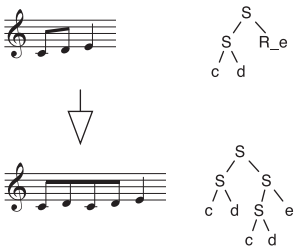

図 3: Example of a Recursive Terminal Node

Increase 操作は前述の Recursive Terminal Node の働きを突然変異として実装したものである．図 4 は Increase 操作の例を示している．この操作を適用されるノードはランダムに選択される．

Decrease は Increase の反対の操作である．ランダムに選択された非終端ノードをその直下の終端ノードで置き換える．

また，遺伝操作を適用する個体は，Fitness-proportional Selection によって決められる．

3. インターフェイスと作曲プロセス

3.1 マルチフィールド・ユーザインターフェイス

CACIE では，[Unemi 98] で提唱された Multi-Field User Interface の考え方を採用した．

図 5，6 は CACIE のユーザインターフェイスである．

図 5 は，CACIE 操作のメインとなるウィンドウである．ウィンドウ上部は二つに分けられ，左が親集団，右が子集団を表している．子集団は生成された直後に親集団を置き換えずに別のウィンドウに表示される．もし結果が気に入らなければ，ユーザは再度子集団を生成することが可能である．また廃棄する子集団の中に気に入った固体があれば，後述する Genome

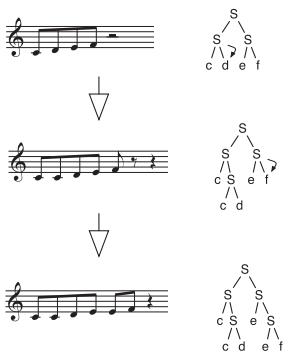

図 4: Increase mutation

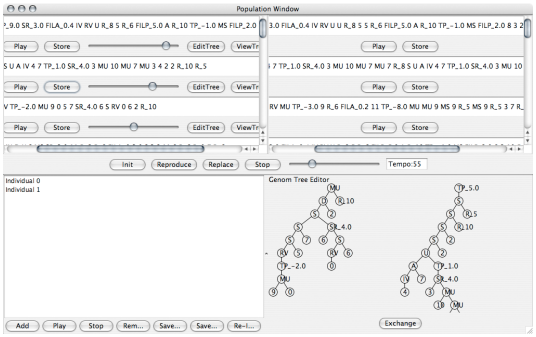

図 5: “CACIE” main interface

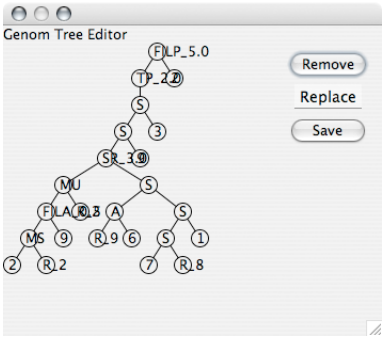

図 6: “CACIE” Tree editor window

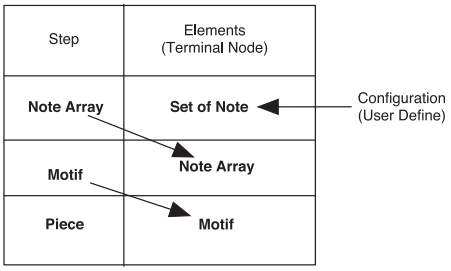

図 7: Multiplex Structure: Phenotypes of a previous step are used for terminal nodes of the next step

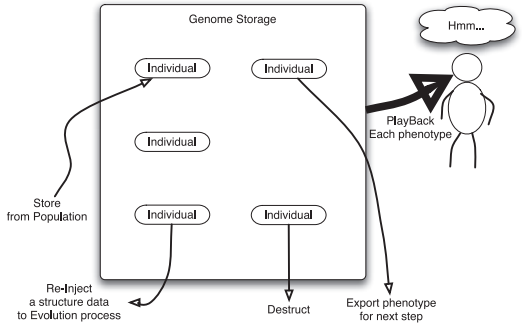

図 8: Genome Storage - Storing the user’s ideas

Storage を用いて親集団の中にその個体を挿入することもできる．

親集団へ適合値を与えるためにはスライダーを使用する．スライダーで適合度を与えれば，状況に応じてユーザが大まかにも細やかにも適合度を与えられるという利便性がでる．これは，Takagi らが示した適合値のレンジが小さいことによるユーザの負担軽減を狙ったものである [Takagi 96a, Takagi 96b]．内部的な適合値のレンジは 100 段階としたので，細かな差異が必要になった場合には別のフィールドに直接数値を入力することが可能である．

図 5 の左下部は，後述する Genome Storage である．

3.2 遺伝子の手動編集

また，図 6 は，遺伝子ツリーを手動で編集することが可能なウィンドウである．また図 5 の右下部では，同様に手動による交叉を行うことができる．

対話型進化計算を作曲に用いた場合，ユーザの好みに近い個体が発生した時点から，完全にユーザの好みに合致した状態に収束するまでの時間がながく，非常にユーザにストレスを与える．この手動編集の手法を用いる事で，ユーザ負担を著しく減らすことができる．また最初からユーザ定義の遺伝子ツリーを挿入し，それをもとに進化させることも可能になっている．

3.21 作曲プロセスの複合構造化CACIE における楽曲の生成は，クラシック音楽の作曲の過程をモデル化した Multiplex 構造によって行われる．典型的な古典クラシック音楽の作曲の過程はいくつかに分けられる．最初はごく短い音列を作成する段階，次に音列を用いたモチーフとその変奏を作成する段階，最後は生成されたモチーフや変奏を変形しながら一つの曲として組み合わせる段階である．筆者らはこのクラシック音楽の複合構造的な作曲方法を取り込んだ．CACIE では，ユーザは一度の探索で曲を生成するのではなく，段階的に曲を生成していく．この複合構造は，それぞれの探索の結果を MIDI をベースとしたイベントリストとしてエクスポートし，次回の探索の終端ノードとして初期集団生成時にインクルードするという実装法で実現している．

3.22 遺伝子ストレージ図 5 の下部に見えるウィンドウは Genome Storage と呼ばれる．この Genome Storage は進化の過程に対するユーザの積極的な関与を実現するために実装した．ユーザはいつでも親集団や子集団から好きな個体を選択し，その染色体と phenotype を Genome Storage に保管 (Store) することが可能である．また，Genome Storage に蓄えられた個体をいつでも親集団へ挿入 (Re-inject) することが可能である．

Multiplex 構造はこの Genome Storage の機能を用いて実現されており，作曲の各ステップの流れは図 9 が示すような流れになる．

4. 実験結果

遺伝子表現とシステムの妥当性を検証するために，実際にモチーフと楽曲の合成を行った．結果の楽譜と SMF は Web サイト上に提示した*1 ．

検証は，三段階にわけて行った．まず始めに，ツリー構造と Recursive Temrinal Node，特殊な遺伝子操作を確かめるために，Genome Storage を使わずに極短い音列の生成を行った．ツリー構造が変化を伴った繰り返し構造を生成するように成長していることを確認した．

次に，対話型進化論的計算のユーザインタフェイスの妥当性

*1 http://www.iba.k.u-tokyo.ac.jp/~dando/public/projects/ecmusic/cacie/results.html

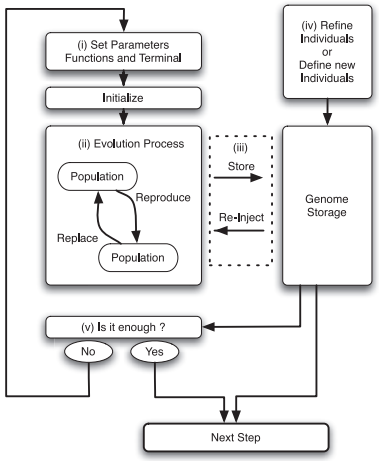

図 9: Flow chart of evolutionary process: User should collect materials for the next step, trying to apply various combinations of parameters, functions and terminal nodes.

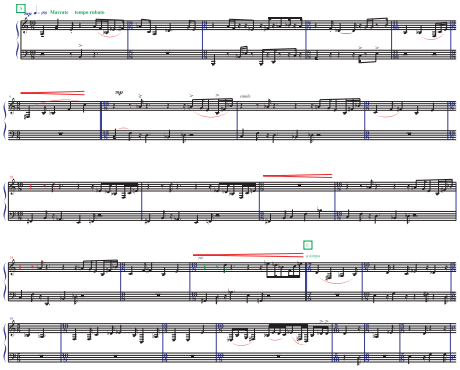

図 10: A part of piano miniture.

を確かめるために Genome Storage を利用したモチーフのと変奏の生成を行った．Genome Storage を使用し，結果の進化への再挿入などを行うと，少ない集団サイズにもかかわらず，音楽的に意味のある変奏を効率よく生成できることが確認できた．

最後に，複合構造を検証するために，比較的長い曲の合成を行った．ここでは，前記の再挿入などの手法の他に，ユーザの手動による染色体の修正も行い，音楽的な構造を容易に生成できることを確認した．

それらの結果を踏まえ，最終的にピアノの現代曲的な小品の合成を行った．得られた結果には，クラシック音楽に見られる典型的な繰り返し構造が大きなレベルから小さなレベルまで見られた．曲の全体的な構造は A-B-C-D-A'-B'-E という形になっている．また各セクションの中にも，モチーフの変奏を生成しながら繰り返しを行う旋律が多く見られた．

また合成した曲を職業演奏家に演奏してもらった所，比較的高い評価をえることが出来ている．この結果を元に，このような専門職とのコラボレーションを現在行っている．

生成された曲のフルスコアと実際の演奏家に演奏してもらったデータは，前述の Web サイトに提示した．

5. おわりに

本稿では，伝統的な作曲家が使うための対話型進化論的計算についての考察と新たなシステムの構築の報告を行った．筆者らが新たに構築したシステムは，クラシック音楽のアナリーゼ手法に基づいた遺伝子表現と作曲手法に基づいた生成プロセスを基本としている．結果として，伝統的な音楽表現を含んだ比較的長い曲の合成に成功した．

参考文献

[Burton 99] Burton, A. R. and Vladimirova, T.: Generation of Musical Sequences with Genetic Techniques, *Computer Music Journal*, Vol. 24, No. 4, pp. 59–73 (1999)

[Dahlstedt 04] Dahlstedt, P. and Nordahl, M. G.: Augumented Creativity: Evolution of musical score material (2004)

[Dawkins 86] Dawkins, R.: *The Blind Watchmaker*, Longman, Essex (1986)

[Johanson 98] Johanson, B. E. and Poli, R.: GP-Music: An Interactive Genetic Programming System for Music Generation with Automated Fitness Raters, Technical Report CSRP-98-13, School of Computer Science, The University of Birmingham (98)

[Laine 94] Laine, P. and Kuuskankare, M.: Genetic Algorithms in Musical Style Oriented Generation, in *Proceedings of First IEEE Conference on Evolutionary Computation*, pp. 858–861, Washington D.C. (1994), IEEE

[Putnam 94] Putnam, J. B.: Genetic Programming of Music, Unpublished manuscript. Socorro, NM: New Mexico Institute of Mining and Technology (1994)

[Takagi 96a] Takagi, H. and Ohya, K.: Discrete Fitness Values for Improving the Human Interface in an Interactive GA, in *Proceedings of IEEE 3rd International Conference on Evolutionary Computation (ICEC'96)*, pp. 109–112, Nagoya (1996), IEEE

[Takagi 96b] Takagi, H., Ohya, K., and Ohsaki, M.: Improvement of Input Interface for Interactive GA and its Evaluation, in *Proceedings of International Conference on Soft Computing (IIZIKA'96)*, pp. 490–493, Iizuka (1996), World Scientific

[Tokui 00] Tokui, N. and Iba, H.: Music Composition with Interactive Evolutionary Computation, in *Proceedings of 3rd International Conference on Generative Art (GA2000)* (2000)

[Unemi 98] Unemi, T.: A Design of Multi-Field User Interface for Simulated Breeding, in *Proceedings of 3rd Asian Fuzzy System Symposium: The Korea Fuzzy Logic and Intelligent Systems* (1998)