OPTISIA: An Evolutionary Approach to Parameter Optimisation in a Family of Point-Set Pattern-Discovery Algorithms

Viktor Schmuck(✉) and David Meredith

Aalborg University, Aalborg, Denmark
{vsch,dave}@create.aau.dk

Abstract. We propose a genetic algorithm (GA), OPTISIA, for efficiently finding optimal parameter combinations when running OMNISIA [15], a program that implements a family of analysis and compression algorithms based on the SIA point-set pattern discovery algorithm [20]. The GA, when given a point-set representation of a piece of music as input, runs OMNISIA multiple times, attempting to evolve a combination of parameter values that achieves the highest compression factor on the input piece. When evaluated on two musicological tasks, the system consistently selected well-performing parameters for Forth's algorithm [6] compared to combinations found in published evaluations on the same musicological tasks.

Keywords: Pattern discovery · Genetic algorithm · Parameter optimization · Music analysis · COSIATEC · OMNISIA · Geometric algorithms · Forth's algorithm · Point sets

1 Introduction

Genetic algorithms (GAs) provide a biologically inspired, evolutionary approach to optimisation problems [9]. Previous work suggests that GAs can provide a time-efficient and custom-fit solution when finding optimal parameter combinations in a variety of contexts [7,14]. We propose a decimal-encoding-based GA for efficiently finding optimal parameter combinations when running OMNISIA [15], a program that implements a family of analysis and compression algorithms based on the SIA point-set pattern discovery algorithm [20]. OMNISIA provides implementations of three compression-based pattern mining algorithms, COSIATEC [21], SIATECCompress [18], and Forth's algorithm [6]. Moreover, it allows each of these algorithms to be run with a wide range of options, such as replacing SIA with SIAR [3] or SIACT [5] or using chromatic or morphetic pitch representations [16,17].

In this paper, we present OPTISIA, a GA-based algorithm that runs OMNISIA on a point-set representation of a piece of music multiple times, evolving a combination of parameter values that optimise the achieved compression

© Springer Nature Switzerland AG 2020
P. Cellier and K. Driessens (Eds.): ECML PKDD 2019 Workshops, CCIS 1168, pp. 509–516, 2020.
https://doi.org/10.1007/978-3-030-43887-6_45

factor. The output of this evolutionary process is the analysis of the input piece generated by the particular parameter value combination represented by the simplest chromosome in the final generation that achieves the maximum compression factor on that input piece.

The choice of compression factor as our fitness function is motivated by the widely accepted principle that the shortest (lossless) encodings of a data object represent the best explanations for that object. This parsimony principle (a.k.a. "Ockham's razor") can be traced back to antiquity and has been formalized in more recent times in various ways, including the MDL principle [22] and Kolmogorov's structure function [24]. A number of recent studies in music information retrieval have demonstrated the potential of using the parsimony principle for classification and clustering tasks [2,12,19] and thematic/motivic analysis [18]. We have tested our new approach on two music-analytical tasks: (1) discovering subject and countersubject entries in the fugues of the first book of J. S. Bach's *Das Wohltemperirte Clavier* [8]; and (2) discovering themes and sections in the polyphonic version of the JKU Patterns Development Database [4].

2 Previous Work on Parameter Tuning with Genetic Algorithms

Genetic algorithms present a biologically inspired approach to optimisation problems based on evolution [9]. This approach creates a map of parameters which can be used as possible permutations of genes. To create a population, chromosomes are formed by chaining genes. The population is advanced by computing a fitness score for each parameter combination (chromosome) to perform selection. There are a number of widely used selection methods such as fitness proportionate, roulette-wheel sampling, and elitist selection [10,13]. When the number of chromosomes is reduced to a desired group (parent population), their chromosomes are recombined in pairs (crossover) to produce members for the next generation. Mutation might also be applied to chromosomes, resulting in a potential value change on one or more of the genes. The randomly selected value from the gene types is often allowed to hold its previous value, resulting in no mutation. Genetic algorithms can, with the application of evolutionary principles, evolve optimal or near-optimal parameter combinations over generations [7]. Genetic algorithms are able to reduce the time required for parameter optimisation [7]. Moreover, they allow the encoding of interval values. This is illustrated in [14], where the proposed decimal encoding of nominal and interval parameters results in shorter chromosomes and the accuracy reaches that obtained with binary encoding. Due to shorter chromosomes the search efficiency of the approach is increased relative to other encoding methods.

3 OMNISIA

OMNISIA [15] is a Java program that implements a family of analysis and compression algorithms based on the SIA point-set pattern discovery algorithm [20].

OMNISIA provides implementations of three compression-based pattern discovery algorithms, COSIATEC [21], SIATECCompress [18], and Forth's algorithm [6]. Moreover, it allows each of these algorithms to be run with a wide range of optional parameter settings. The program has been used in a number of previous studies on music analysis and generation [1,11,19].

The descriptions of the various algorithms implemented in OMNISIA and their parameters are given in the original papers describing the algorithms and summarised in [19]. Figure 1 illustrates the effect of some of these switches on the output generated by OMNISIA for the C minor Prelude (BWV 871) from Book 2 of J. S. Bach's *Das Wohltemperirte Clavier*.

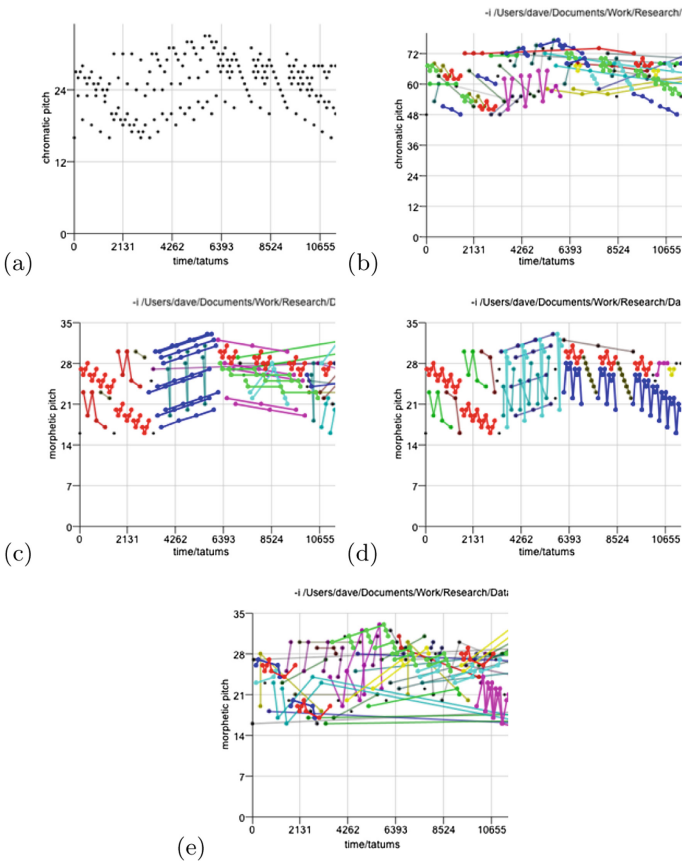

Fig. 1. Example outputs of the OMNISIA program. (a) Point set representation of the prelude from BWV871 given as input. (b) Output generated by COSIATEC using chromatic pitch. (c) Output generated by COSIATEC using morphetic pitch with -d switch selected. (d) Output generated using morphetic pitch and compactness trawler (-ct switch). (e) Output generated when SIA is replaced with SIAR.

4 OPTISIA: An Evolutionary Approach to Parameter Optimisation in OMNISIA

To solve the problem of optimising parameters for the OMNISIA program, the various compression-based pattern mining algorithms and their options were mapped. OMNISIA can use three base algorithms: COSIATEC (`COS`) [21], SIATECCompress (`SCo`) [18], and Forth's algorithm (`FoA`) [6]. The first two require 7 switches, while Forth's algorithm requires 11. The option values of switches were sorted, based on them being nominal (enabled/disabled) or interval (a range of values). The interval values were mapped to ordinal ones to shrink the search space of the optimisation (Table 1).

Table 1. The outline of option switch prefixes, their nominal–interval distinction and presence when using different base algorithms to run OMNISIA.

Switches		Base algorithms	
Switch prefixes	Nominal (N) or Interval (I)	COSIATEC, SIATECCompress	Forth
-d	N	X	X
-ct	N	X	X
-cta	I	X	X
-ctb	I	X	X
-rsd	N	X	X
-r	I	X	X
-rrt	N	X	X
-crlow	I		X
-comlow	I		X
-cmin	I		X
-bbcomp	N		X

To design a gene pool for each element of the chromosomes, the base algorithm options need to be encoded. Following the work of Liu and Wang [14] the values are decimal-encoded. Therefore, gene values range from 0 to 9 instead of multiple genes describing a single parameter that has more than two types. Decimal encoding was chosen to minimise the number of genes the algorithm has to handle during evolution, lowering the required population size and generation count, and consequently the running time of the algorithm.

Some options in OMNISIA are dependent on each other. Therefore, when creating chromosomes, if the nominal values of '-ct' (compactness trawling) [5] or '-rsd' (r superdiagonals) [3] are not set to 'True', the dependent parameters of '-cta' (minimum compactness of trawled patterns), '-ctb' (minimum size of trawled patterns), and '-r' (number of superdiagonals used in SIAR) should not hold values either. This relation between genes was respected during the crossover and mutation operations of the GA. In a chromosome, if the dependent values were not set beforehand, they were initialised at random to complete it.

To create the first population of chromosomes, genes were selected in randomised combinations, discarding repeated ones, therefore no chromosomes were the same at the start. The population size was chosen based on the amount of genes required to encode all chromosomes (7 for `COS` and `SCo`, and 11 for `FoA`). Due to the selection and recombination described below, it is more beneficial to choose population counts divisible by 3. Taking the calculated population sizes based on Gutowski's [9, p. 198] inequality, the previously described divisibility, and the observed population counts [14,23] into account, we tested the execution time and fitness on a single piece from the Fugues database [8] with population sizes of 12, 15 and 18 for `COS` and `SCo`, and 18, 21 and 24 for `FoA`. If the achieved maximum fitness scores were identical in two cases, the lower execution time was used to set the population size, resulting in 12 for `COS` and `SCo`, and 21 for `FoA`.

Fitness scores were acquired by running the parameter combinations and retrieving the resulting compression factors. To create subsequent generations, 1/3 of the population was kept with elitism-based selection. The genes of these parent-chromosomes were recombined in randomly selected pairs to create 4 offspring chromosomes each, so that the new generation could reach the set population size. An illustration of the selection and recombination can be seen in Fig. 2. Finally, each gene within the offspring chromosomes had its mutation chance set to $100/C_{\text{len}}$ where C_{len} is chromosome length. Genes undergoing mutation were allowed to take their previous values at random.

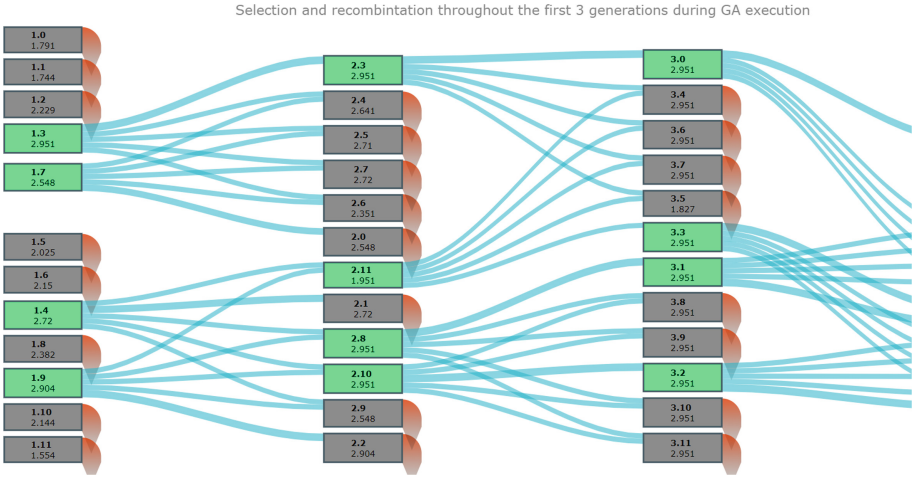

Fig. 2. The figure shows the first 3 generations of chromosomes and their calculated fitness scores. Green cells show parent chromosomes. Blue lines show the recombination. (Color figure online)

The GA optimisation was terminated if the fitness failed to improve (stagnated) for k generations after a minimum of g generations, where $k = 15$ and $g = 30$ for `COS` and `SCo`, and $k = 30$ and $g = 40$ for `FoA`, following the generation

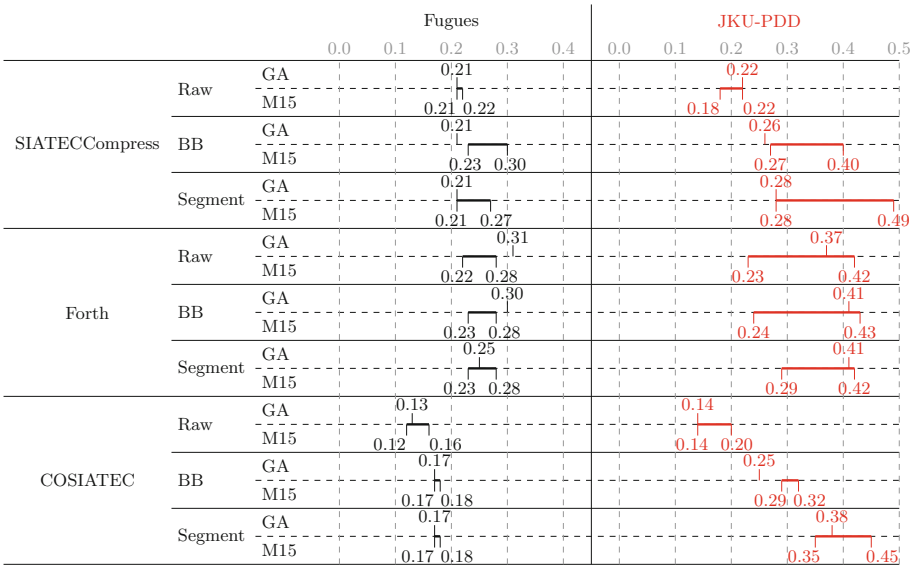

Fig. 3. Results of using OPTISIA to discover subjects and countersubjects in the fugues of the first book of J. S. Bach's *Das Wohltemperirte Clavier* (left-hand side of figure, in black); and discover repeated themes and sections in the JKU Patterns Development Database (right-hand side of figure, in red). Values are for three-layer F_1 score [19, pp. 256–259]. See main text for details. (Color figure online)

count estimation proposed in [9, p. 198]. In most cases, the parameter optimisation of each piece was stopped by the previously described early-termination mechanism.

5 Evaluation

We evaluated our approach on two music-analytical tasks: (1) discovering subject and countersubject entries in the fugues of the first book of J. S. Bach's *Das Wohltemperirte Clavier* [8]; and (2) discovering themes and sections in the polyphonic version of the JKU Patterns Development Database [4]. Figure 3 summarizes the results obtained. In each of the two experiments, the best-compressing chromosome discovered by the GA for each of the three basic algorithms (`COS`, `SCo` and `FoA`) was run in "Raw", "BB" and "Segment" mode (see [19] for an explanation of these terms). The rows headed "GA" in Fig. 3 show the three-layer F_1 (TLF1) scores [19, pp. 256–259] obtained using these nine algorithm–mode combinations on the two experiments. The rows headed "M15" in Fig. 3 show, for each experiment, for each basic algorithm and for each mode, the range of TLF1 scores obtained for that experiment in [19] for parameter combinations using the same algorithm and mode.

For FoA, Fig. 3 shows that, for all modes, the chromosome automatically selected by the GA performed well compared with the best of the Forth chromosomes tested in [19] on both the JKU-PDD and the fugues database. However, for COS and SCo, the GA typically selected a chromosome that performed poorly relative to previously tested parameter combinations. Indeed, in several cases, the GA-selected chromosome performs worse than any of the previously tested parameter combinations for the given algorithm and mode. We speculate that the poorer performance of our GA-based approach on COS and SCo, is at least partly due to the lower gene count in the chromosomes for these algorithms. Perhaps this problem could be mitigated by increasing the probability of mutation in order to avoid the population converging on a relatively poorly-performing, but locally optimal, parameter combination.

6 Conclusion and Suggestions for Future Work

We used a GA with compression factor as a fitness function to evolve parameter combinations for the SIATEC-based analysis algorithms implemented in the OMNISIA point-set analysis program. When the approach was evaluated on two musicological pattern discovery tasks, it was found that it consistently selected a high performing parameter value combination for Forth's algorithm [6], but relatively poorly-performing parameter combinations for COSIATEC [21] and SIATECCompress [18]. It may be possible to improve the genetic algorithm's efficiency, and possibly increase the achieved compression factor, by mapping the base algorithm options that were mapped to hold ordinal values to interval ones instead. In addition, the effect of using fitness proportionate selection should be investigated. Lastly, it can be hypothesized that this approach, as opposed to elitist selection, would require more computation time, but it would be less prone to stagnation despite the presence of local maxima in the space defined by the fitness function.

References

1. Boot, P., Volk, A., de Haas, W.B.: Evaluating the role of repeated patterns in folk song classification and compression. J. New Music Res. **45**(3), 223–238 (2016)
2. Cilibrasi, R., Vitányi, P.M.B., de Wolf, R.: Algorithmic clustering of music based on string compression. Comput. Music J. **28**(4), 49–67 (2004)
3. Collins, T.: Improved methods for pattern discovery in music, with applications in automated stylistic composition. Ph.D. thesis, Faculty of Mathematics, Computing and Technology, The Open University, Milton Keynes (2011)
4. Collins, T.: JKU Patterns Development Database (2013). https://dl.dropbox.com/u/11997856/JKU/JKUPDD-Aug2013.zip
5. Collins, T., Thurlow, J., Laney, R., Willis, A., Garthwaite, P.H.: A comparative evaluation of algorithms for discovering translational patterns in Baroque keyboard works. In: 11th International Society for Music Information Retrieval Conference (ISMIR 2010), pp. 3–8 (2010)

6. Forth, J.C.: Cognitively-motivated geometric methods of pattern discovery and models of similarity in music. Ph.D. thesis, Department of Computing, Goldsmiths, University of London (2012)
7. Francescomarino, C.D., et al.: Genetic algorithms for hyperparameter optimization in predictive business process monitoring. Inf. Syst. **74**, 67–83 (2018)
8. Giraud, M., Groult, R., Levé, F.: Truth file for the analysis of Bach and Shostakovich fugues (2013/12/27 version) (2013). http://www.algomus.fr/truth/fugues.truth.2013.12
9. Gutowski, M.W.: Biology, physics, small worlds and genetic algorithms. In: Shannon, S. (ed.) Leading Edge Computer Science Research, pp. 165–218. Nova Science Publishers Inc., New York (2005)
10. Hancock, P.J.B.: An empirical comparison of selection methods in evolutionary algorithms. In: Fogarty, T.C. (ed.) AISB EC 1994. LNCS, vol. 865, pp. 80–94. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-58483-8_7
11. Herremans, D., Chew, E.: MorpheuS: generating structured music with constrained patterns and tension. IEEE Trans. Affect. Comput. (2017). https://doi.org/10.1109/TAFFC.2017.2737984
12. Hillewaere, R., Manderick, B., Conklin, D.: String quartet classification with monophonic models. In: 11th International Society for Music Information Retrieval Conference (ISMIR 2010), Utrecht, The Netherlands, pp. 537–542 (2010)
13. Lipowski, A., Lipowska, D.: Roulette-wheel selection via stochastic acceptance. CoRR abs/1109.3627 (2011). http://arxiv.org/abs/1109.3627
14. Liu, Y., Wang, C.: A modified genetic algorithm based optimisation of milling parameters. Int. J. Adv. Manuf. Technol. **15**(11), 796–799 (1999)
15. Meredith, D.: OMNISIA, software. http://www.titanmusic.com/software/omnisia/OMNISIA20160926.zip
16. Meredith, D.: The ps13 pitch spelling algorithm. J. New Music Res. **35**(2), 121–159 (2006)
17. Meredith, D.: Computing pitch names in tonal music: a comparative analysis of pitch spelling algorithms. Ph.D. thesis, Faculty of Music, University of Oxford (2007)
18. Meredith, D.: COSIATEC and SIATECCompress: pattern discovery by geometric compression. In: MIREX 2013, Competition on Discovery of Repeated Themes & Sections (2013). https://www.music-ir.org/mirex/abstracts/2013/DM10.pdf
19. Meredith, D.: Music analysis and point-set compression. J. New Music Res. **44**(3), 245–270 (2015)
20. Meredith, D., Lemström, K., Wiggins, G.A.: Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. J. New Music Res. **31**(4), 321–345 (2002)
21. Meredith, D., Lemström, K., Wiggins, G.A.: Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. In: Cambridge Music Processing Colloquium (2003). http://www.titanmusic.com/papers/public/cmpc2003.pdf
22. Rissanen, J.: Modeling by shortest data description. Automatica **14**(5), 465–471 (1978)
23. Saini, L.M., Aggarwal, S.K., Kumar, A.: Parameter optimisation using genetic algorithm for support vector machine-based price-forecasting model in national electricity market. IET Gener. Transm. Distrib. **4**(1), 36–49 (2010)
24. Vereshchagin, N.K., Vitányi, P.M.B.: Kolmogorov's structure functions and model selection. IEEE Trans. Inf. Theory **50**(12), 3265–3290 (2004)