A Caching Genetic Algorithm for Spectral Breakpoint Matching

Jonathan Mohr[1] and Xiaobo Li[2]

[1] Augustana University College, Camrose, Alberta, Canada T4V 2R3
mohrj@augustana.ca, http://www.augustana.ca/~mohrj/
[2] University of Alberta, Edmonton, Alberta, Canada T6G 2M7
li@cs.ualberta.ca, http://www.cs.ualberta.ca/~li/

Abstract. Two methods were evaluated for performing spectral breakpoint matching: a multi-level pruned exhaustive search and a genetic algorithm. The GA found matches about as good as those found by pruned search, but savings in time were realized only if the objective function cached the results of previous evaluations.

1 Introduction

Multiple wavetable interpolation synthesis [1] is a form of music analysis/synthesis in which a recorded sound is reduced to a set of breakpoints by piecewise linear approximation of the spectral envelopes of its harmonics. The spectrum at each breakpoint is then matched by determining weightings for a small number of wavetables selected from a *wavetable bank*, and the sound is resynthesized using multiple wavetable additive synthesis by interpolating between the weightings for each wavetable at consecutive breakpoints.

2 Spectral Breakpoint Matching

Given a particular wavetable bank, a set of breakpoint data representing a particular tone, and the number of oscillators (N) to be used in resynthesis, our breakpoint-matching algorithm [2] selects at most N wavetables from the bank of basis spectra that, in weighted combination, best match the spectrum at each breakpoint according to some error measure.

In the first stage of the algorithm, an initial match is found for each breakpoint. In a subsequent optimization stage, a weighted wavetable is assigned to each available oscillator at each breakpoint such that the overall error is minimized, taking into account the need to fade a wavetable in or out when it begins or ceases to be used [3].

Two methods were evaluated for finding initial matches in the first stage: a multi-level pruned search and a genetic algorithm (GA). It was found that a "3+1" search—an exhaustive search for the best 3-wavetable matches, augmented with a fourth wavetable by a second-level search—executes about an order of magnitude faster than an exhaustive search for a match of size 4, yet yields about the same or better error rates, on average, after optimization [4].

When a genetic algorithm was used as a first-level search in combination with an exhaustive second-level search, the GA found matches about as good as those found by exhaustive or pruned search; however, savings in time were only realized relative to the larger pruned searches, and then only if the objective function cached the results of previous calls to the evaluation functions.

3 A Caching Genetic Algorithm

Caching was implemented by introducing two mappings as static data members of the objective function so that the contents of the mappings would be preserved across the evaluations of all individuals in all generations. The first maps from sets of wavetables to the results of LUP decompositions, the second, from breakpoint number and wavetable set to the corresponding least-squares solution. When iterating across the multimap, the first map is checked for each wavetable set in the current individual, and LUP decompositions are performed only for those wavetable sets not already in the map; similarly, when iterating across the breakpoints at which a given wavetable set is used in the current individual, the second map is searched for a pre-existing least-squares solution and error level.

One of the configurations that was tested used a population size of 50 and early termination (25 generations to convergence). Without caching, this GA performed about 25% more LUP decompositions and least-squares evaluations than the pruned search to find the initial 3-wavetable matches. With caching, the GA performed only 3% as many calculations as the GA without caching, finishing in 11% of the time.

4 Conclusion

The use of mappings as static data members of the objective function to cache previously calculated results for reuse in evaluating the fitness of all individuals across generations was essential in order to make a GA competitive in time with pruned exhaustive search.

References

1. Horner, A.: Computation and memory tradeoffs with multiple wavetable interpolation. Journal of the Audio Engineering Society **44** (1996) 481–496
2. Mohr, J.: Music Analysis/Synthesis by Optimized Multiple Wavetable Interpolation. PhD thesis, University of Alberta (2002)
3. Serra, M.H., Rubine, D., Dannenberg, R.: Analysis and synthesis of tones by spectral interpolation. Journal of the Audio Engineering Society **38** (1990) 111–128
4. Mohr, J., Li, X.: Computational challenges in multiple wavetable interpolation synthesis. In Sloot, P.M., et al., eds.: Computational Science—ICCS 2003. Number 2657 in Lecture Notes in Computer Science, Springer (2003) 447–456